Faculty of Computing

Universiti Malaysia Pahang

BCS2243 Web Engineering

Semester II 2019/2020

Project Task: Project Report

Project Title: UMP Vehicle Tracking System (UMPVi)

Lab Section: 03A

Group Members:

No.	Photo	Name	Student ID
1		MOHD RAFI RIFFLI BIN RAFIZAL	CA17023
2		MOHD ELFARIZ BIN MOHD IRFAN	CA17005
3		MUHAMMAD KHAIRI ‘AFIFI BIN ABD RAZAK	CA17118
4		MUHAMMAD HILMI BIN ALI	CA17009
5		MUHAMMAD TALHAH BIN MOHAMED NASIR	CA17008

TABLE OF CONTENTS

No	Items	Page
1	Project Question	3
2	Proposed Solution	4
3	Module Distribution	5
4	Entity Relationship Model	6 - 7
5	Access Model	8
6	Presentation Model	9
7	Comparisons with 2 Web Applications	11 - 12
8	Web Interfaces	13 - 19
10	SQL Code	20

PROJECT NAME: UMP Vehicle Tracking System (UMPVi)

The application aims to manage the information of staffs' or students' vehicle in UMP campus. The potential users for this application system are administrator; whom an officer from Unit Keselamatan UMP, UMP staffs (staff) and UMP students (student). All users will have their own username and password to access the application. Each group member who responsible for a module should propose any suitable reporting. The report must implement single table and join table using structured query language (SQL). The basic requirements for UMPVi are:

Module 1: Manage vehicle sticker

Each UMP staff and student is required to apply for yearly vehicle sticker. A copy of proof of vehicle ownerships (vehicle grant) must be attached in the application. The administrator needs to approve the application.

Module 2: Manage QR codes

Upon application approval by administrator, an UMP vehicle sticker with QR code will be generated and printed. The vehicle sticker with QR code must be displayed on the vehicle.

Module 3: Manage traffic violation

A penalty will be issued for each traffic violation in UMP (e.g. parking, no sticker displayed). A staff from Unit Keselamatan will need to scan the vehicle sticker with QR code and insert relevant violation details in UMPVi database. The staff will need to insert their staff identification number and password to verify the violation process.

Module 4: Manage point

The application provides a module to calculate total merit obtained by a staff or student according to their obedient to UMP campus traffic. Unit Keselamatan has introduced a penalty point for traffic violation as given in Table 1. There are also events organised by the department where user able to claim for point. In the end of the year, staff or student able to get a free vehicle sticker for the next year based on the cumulative point.

Position	Penalty point
Cause accident	500
Parking violation	400
No sticker displayed	300
Not wearing seat belt or helmet	200

Table 1: Merit deduction for traffic violation in UMP

Module 6: Manage report and data analytics and login

This module will provide the function of reporting such as student, violation, merit or any suitable reports for the system administrator. A group member who responsible for this module should in charge login module.

PROPOSED SOLUTION

Before starting to develop or design a web application, it is essential to produce a fundamental plan on how our web application will be executed. This is to avoid any confusion during development process since we are dealing with multiple number of classes and modules that need to be integrated successfully to make our web applications working. There might be some time taken to do analysis and modelling even though our web application development process should be agile. But it will worth it when we save our time later by avoiding any wrong problem solving towards our web application.

Basically, there are three (3) modelling specifics in web application engineering and development which are modelling the content level, modelling the hypertext level, and modelling the presentation of our web application. After completing the modelling process, then we can customize our model to be preferable to us as the developer. In this proposal, there will be three modelling specifics that will shows basically on how UMPVi system will work which are the entity relationship model, the access model and finally the presentation model.

UMPVi is basically a web application that are used by UMP communities to manage their vehicle by designing and generating a sticker that need to be displayed on the vehicle. This system will help as a medium for the staffs or students to apply their yearly vehicle sticker by only filling up a form for their application. All the applications need to be approved by the administrator using the same system before a sticker is generated with an embedded unique QR code. This QR code are useful in helping to detect whose vehicle in any event such as receiving penalty. Any penalty received by the user will affect their total point and may lose their eligibility to receive a free vehicle sticker for the next year. This system also able to generate report with complete data analytics which will be useful for the safety department (Unit Keselamatan UMP) to analyse and track the traffic record happening in UMP.

Lastly, there are two (2) web applications that are closely related to UMPVi which are used to get inspiration and closely comparing their functions and modules. The technology used to develop the systems also need to be concerned since they will provide both advantages and disadvantages towards our developed system. For this case, we will inspect the Smart Selangor Parking System and the former UMP Vehicle Registration System with UMPVi that will be developed.

MODULE DISTRIBUTION

There are five (5) modules based on the requirement that need to be designed, developed, and integrated successfully in this UMPVi system. We have decided to distribute the modules among our group member, and we need to be well-prepared and aware on any steps done during this project development to make sure the project executed perfectly. The modules and the person in charge are shown in Table 2 below: -

No	Module	Person In-Charge	ID
1	Manage vehicle sticker	Mohd Rafi Riffli	CA17023
2	Manage QR codes	Mohd Elfariz	CA17005
3	Manage traffic violation	Muhammad Khairi ‘Afifi	CA17118
4	Manage point	Muhammad Hilmi	CA17009
6	Manage report, data analytics and login	Muhammad Talhah	CA17008

Table 2: Module distribution among group members

After deciding on the module development for all group members, there should be a planning on how the web applications should be developed and there were time guidelines that we need to follow and monitor carefully. The technologies and languages that will be used to develop this project are HTML, HTML5, CSS, JavaScript, and PHP with database access and manipulation. There are given time guideline after this proposal were submitted, which are useful for observing the project progress and to avoid any procrastination. In Week 12, we should achieve the milestone where our web should be able to manage data for every module such as insert, update, delete and view. Week 13 are the week to show the progress on the user dashboard and reports, while Week 14 where our prototype is completely done.

ENTITY RELATIONSHIP MODEL

The Entity Relationship Model is a type of model under content model where are designed to transfer information and functional requirements determined by the requirements. It is basically a medium that contains structural elements of a web applications. It is used to analyse all the classes which shows a user-visible entity that interacts with the Web applications and also to shows the basic structure to model the hypertext and presentation model.

The figure 1 below shows the entity relationship model for UMPVi web application. There are basically 7 tables that should be created in the database where each table store different type of data and several tables should consists of joining process of other tables data. The tables are WEB USER, USER, POINTS, STICKER, VEHICLE, PENALTY, and REPORT.

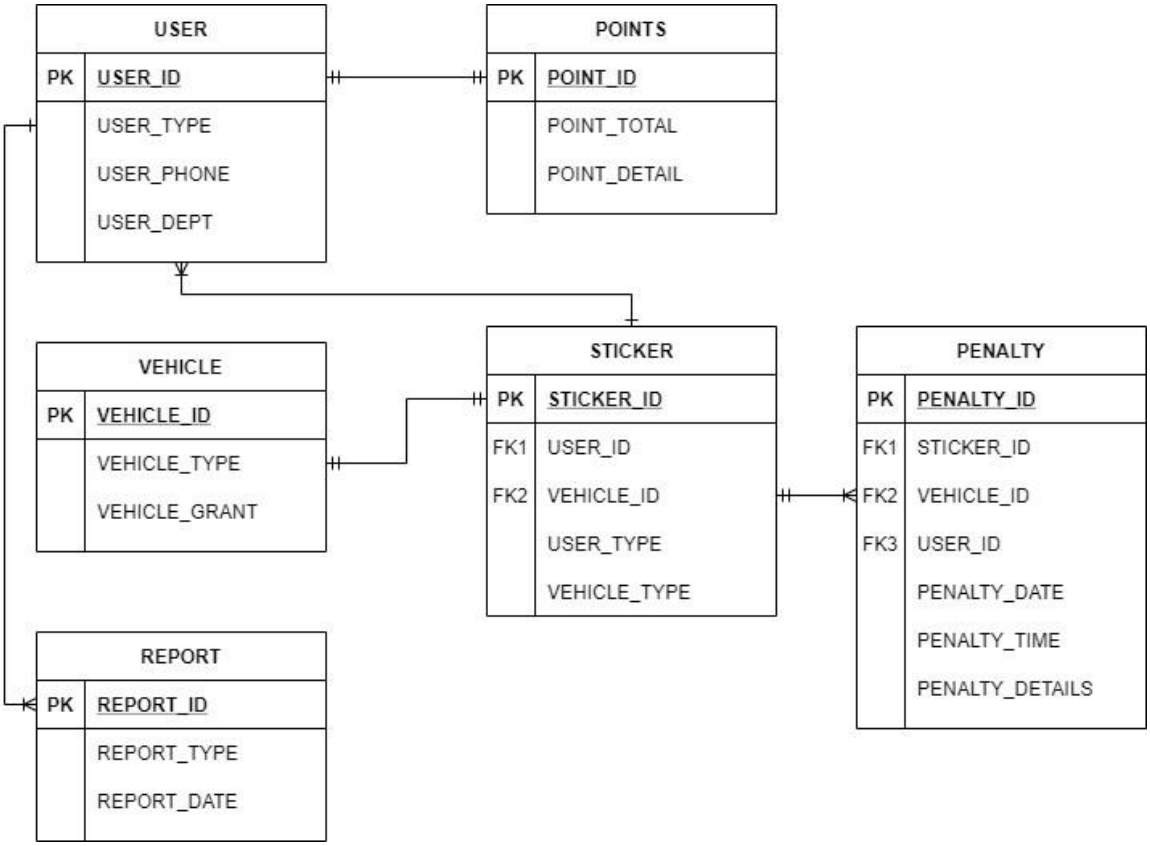


Figure 1: Entity Relationship Model for UMPVi

Figure 2 below show the class diagram of UMPVi:

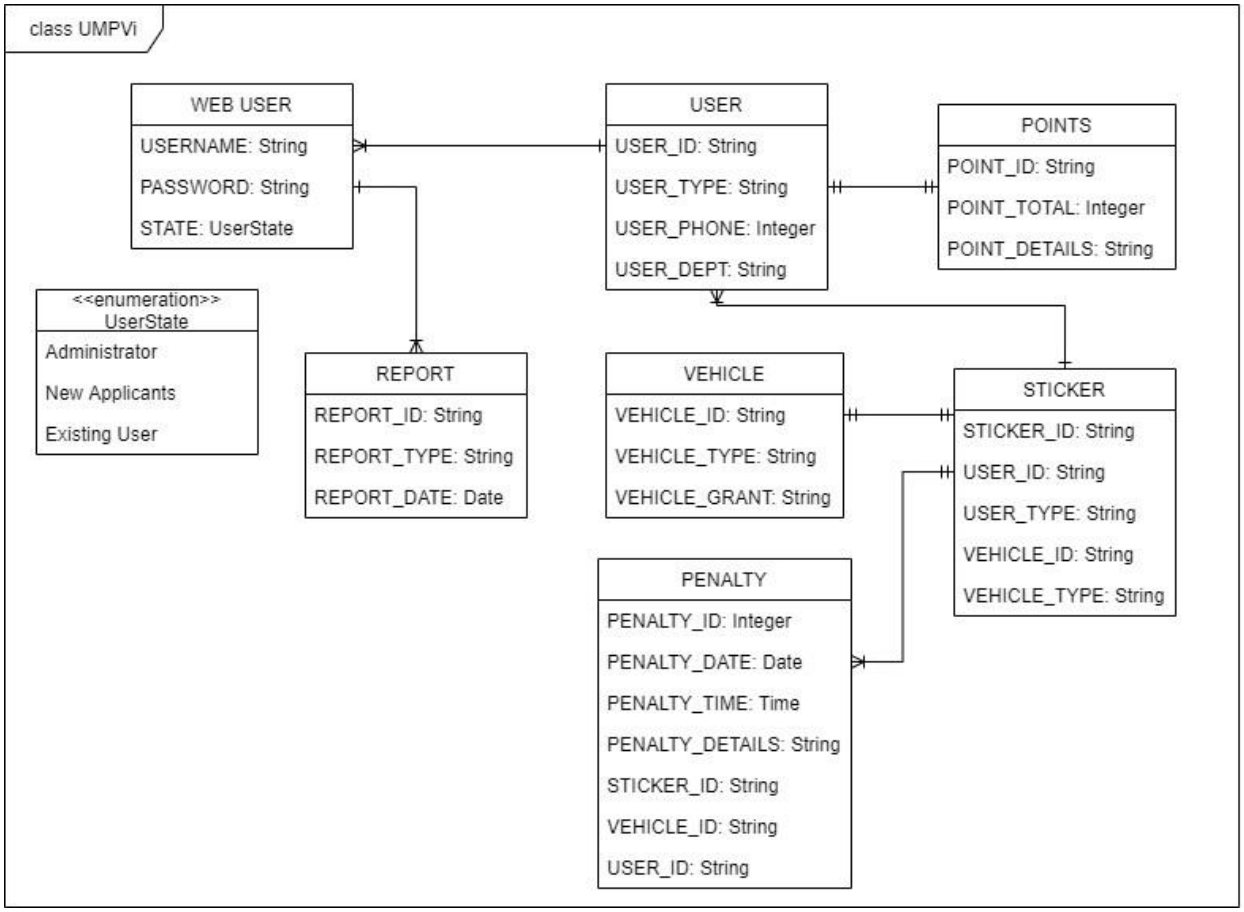


Figure 2: Class Diagram for UMPVi

ACCESS MODEL

Access model is a type of model under the interaction model which are usually considered for the hypertext modelling or in a simple definition known as navigation modelling. The aim of the interaction model is to specify the navigability through the Web application content. There are two type of model that we can use to design interaction model which are hypertext structure model or an access model. But a hypertext structure model is not sufficient to describe how nodes can be reached by navigation. So, we design an access model as a refinement of hypertext structure model.

Figure 2 below shows the access model for UMPVi where it shows how a user can navigate around the web applications to manage their vehicle sticker or to generate report or summon a penalty.

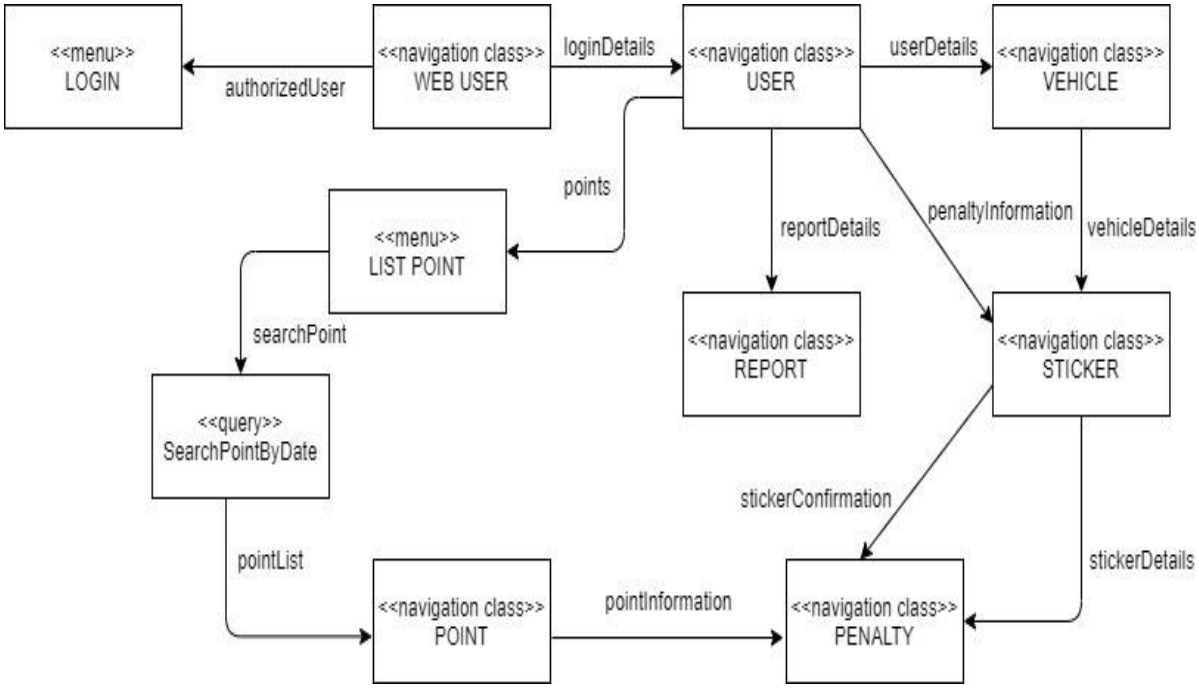


Figure 3: Access Model for UMPVi

PRESENTATION MODEL

Presentation modelling is an essential of the user interface design. The difference between presentation model and the user interface design is we describe the web applications as a visualization unit in the presentation model. Basically, the objective in presentation modelling is to design the structure and the behaviour of the user interface to ensure the interaction with the web application is simple and can be self-explanatory.

The figures below show the presentation model of UMPVi for all the pages that will be developed from the beginning.

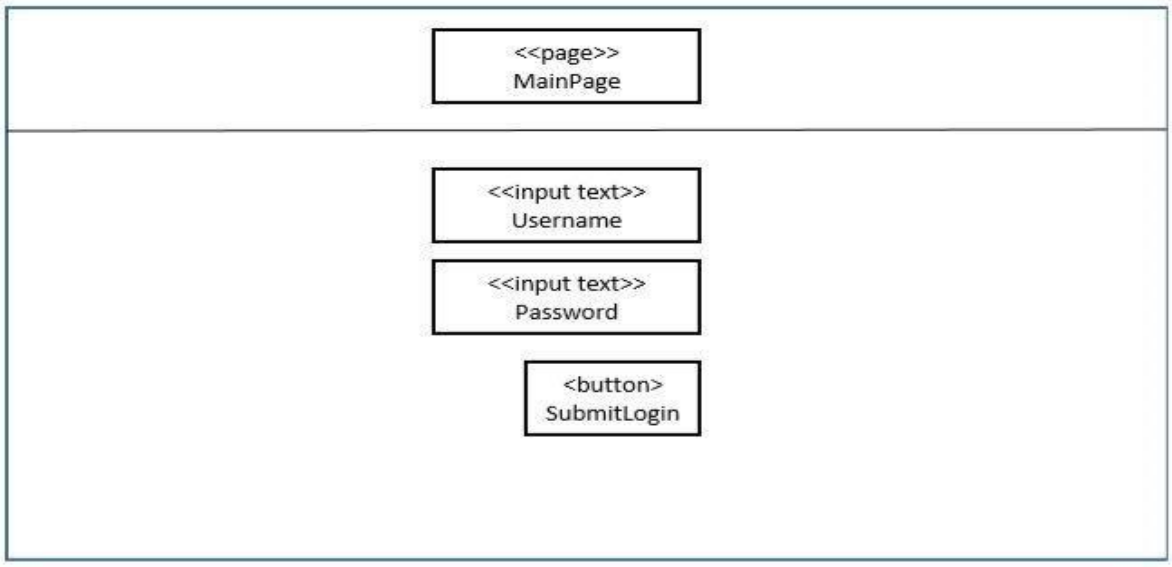


Figure 4: Presentation Model for LOGIN page (Main)

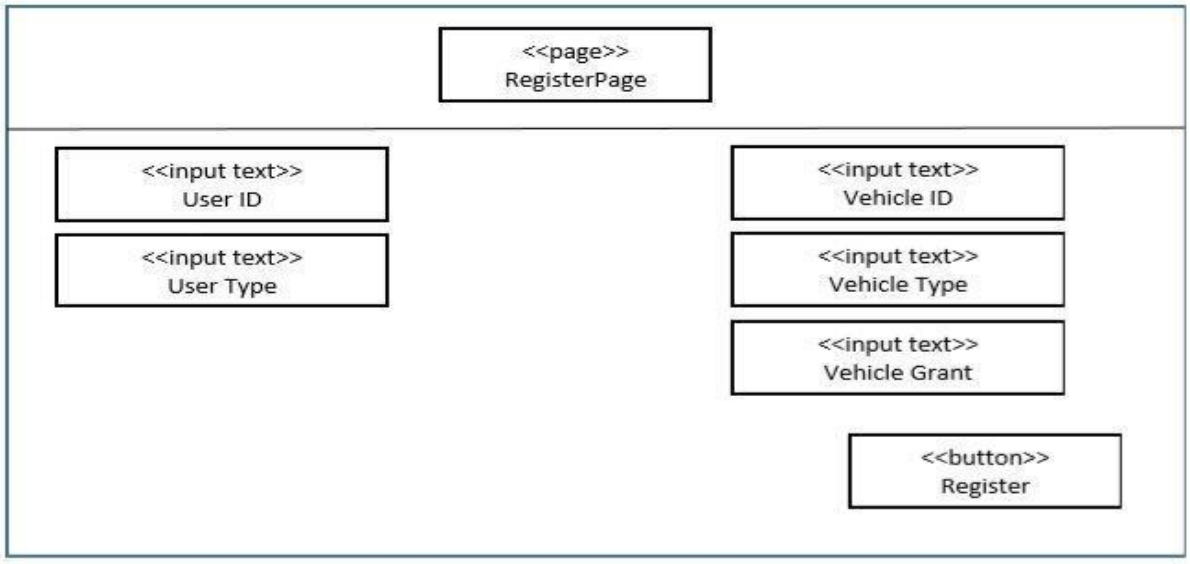


Figure 5: Presentation Model for REGISTER page

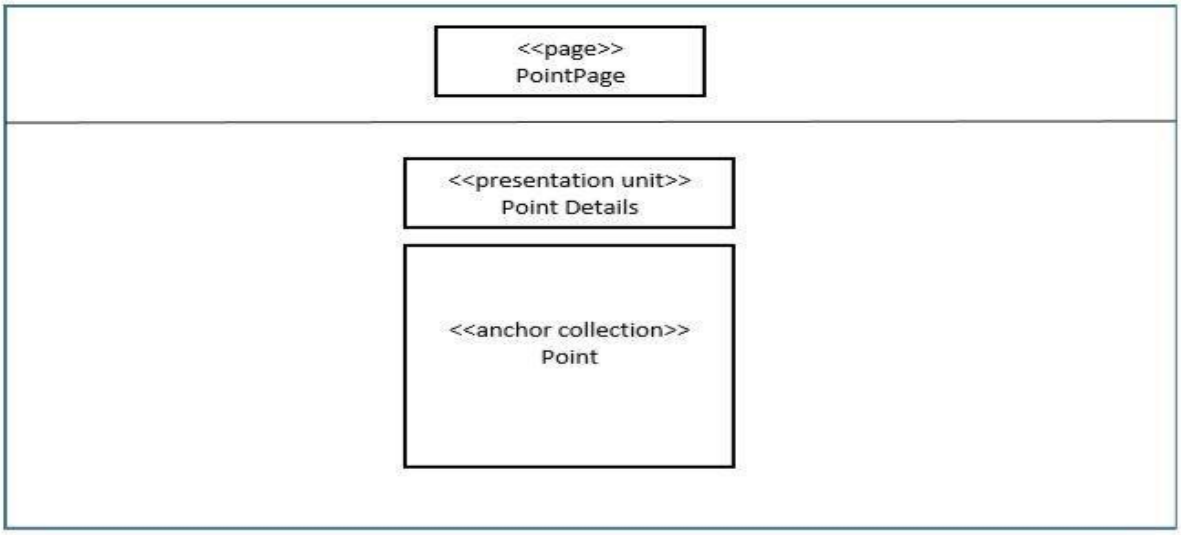


Figure 6: Presentation Model for POINTS page

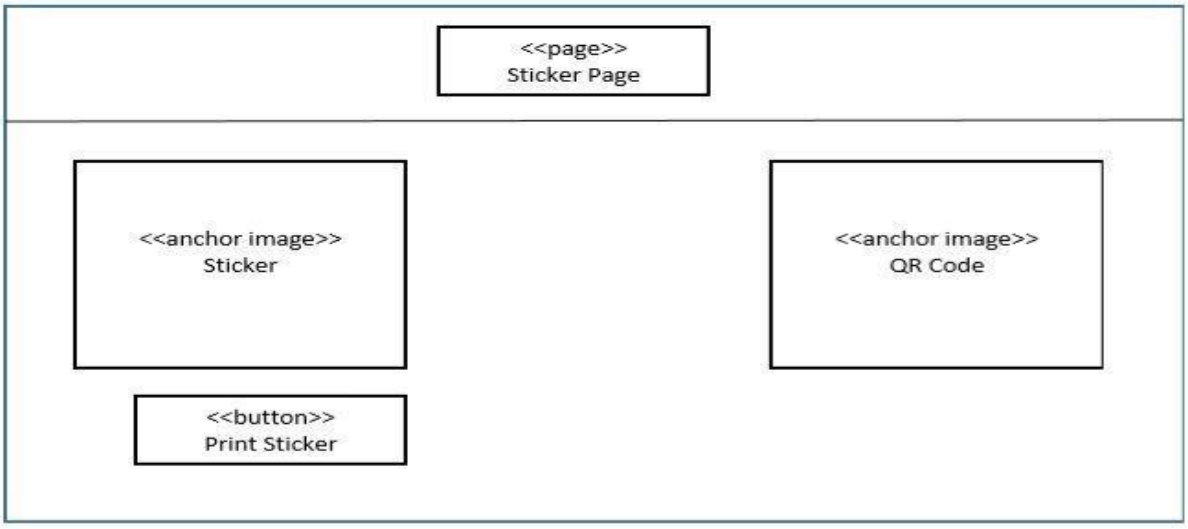


Figure 7: Presentation Model for STICKER page

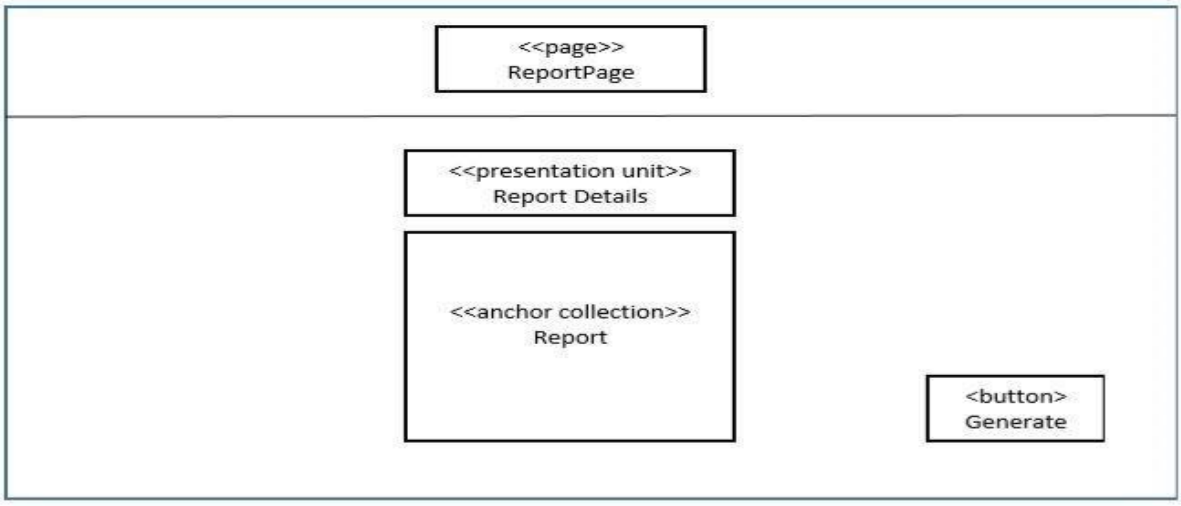


Figure 8: Presentation for REPORT page

COMPARISONS WITH 2 WEB APPLICATIONS

There are two (2) web applications that are closely related to UMPVi which are used to get inspiration and closely comparing their functions and modules. The technology used to develop the systems also need to be concerned since they will provide both advantages and disadvantages towards our developed system. For this case, we will inspect the Smart Selangor Parking System and the Touch N Go RFID Toll Payment to better understanding their features and modules involved to be compared with UMPVi that will be developed.

Table 1 shows the comparison between these 2 applications that are closely related with UMPVi: -

Application Name	**Smart Selangor Parking App**	**Touch N Go Vehicle Toll Payment**
Features	- The Smart Selangor Parking is an application for users to make online parking payment. - Affixed to number plate of the vehicle that have been scan via License Plate Recognition (LPR) by parking officer and this system tied to the Smart Selangor Application.	- The RFID Tag is a sticker that is embedded with a radio frequency chip, and unique to each user. - Affixed to either the windscreen or headlamp of the vehicle, tags are tied to the Touch n Go e Wallet and used as a form of electronic payment for tolls across the country.
Functions	- The users register and activate the Smart Selangor Parking applications with the vehicle number plate. - The data of the users have been stored such as Name, National ID and Vehicle number. - The users can used it for parking payment with selection duration time in Selangor.	- The users register and activate the RFiD in Touch n Go eWallet applications. - The users paste the RFid Tag at windscreen or top left headlamp. - The data of the users have been stored such as Name, National ID and Vehicle number. - The users can used it at any highway located in Peninsular Malaysia.

Advantages	- No longer need to display anything on the vehicle. - Digital receipt stores online and can be printed out when necessary and e-mail receipt as soon as parking payment is made. - Enforcements are all online checking your vehicle number through the server.	No more queuing up, quick, and - secure toll payments. Reload Touch 'n Go eWallet from - anywhere at any time with no reload fees. Transaction history from the app - makes travel claims easier.
Disadvantages	- The accuracy of the License Plate Recognition may decrease upon time and may failed as the license plate font type changed. - Staffs cannot summon a penalty using the system.	- The RFID tag may cannot be detected after long time because of the RFID tag has been tampered. - No penalty can be issued using this system. - No system for point calculation to get free RFID tag for next installation.

Table 3: Comparisons between Smart Selangor and Touch N Go

WEB INTERFACES

Module 1: Manage Vehicle Sticker

User Interface: Applying Sticker

After user login/register, they will be redirected to the home page where they can choose several options from the navigation tab. The sample of the home page interface are shown below:

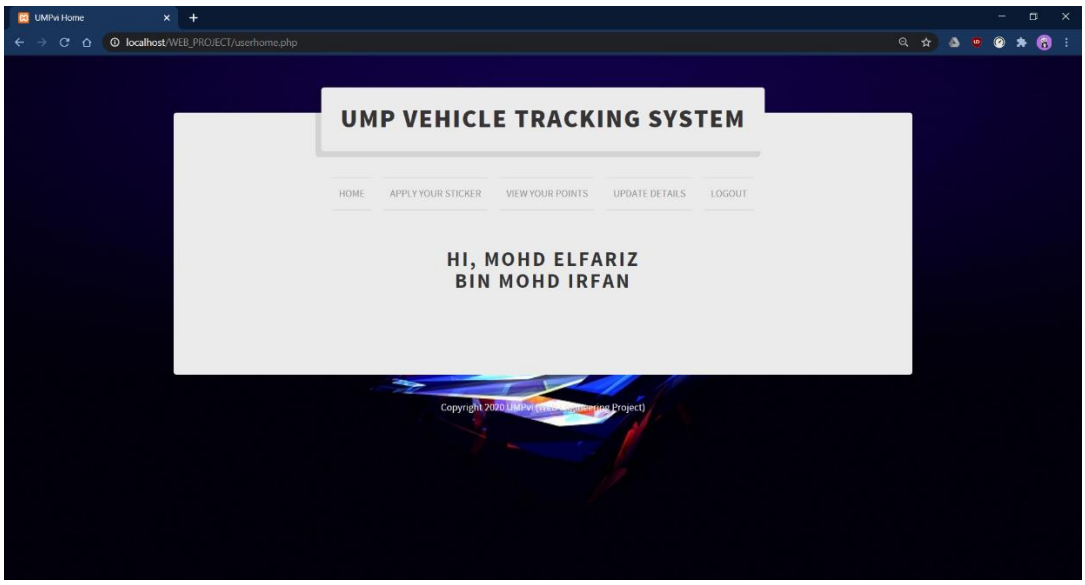
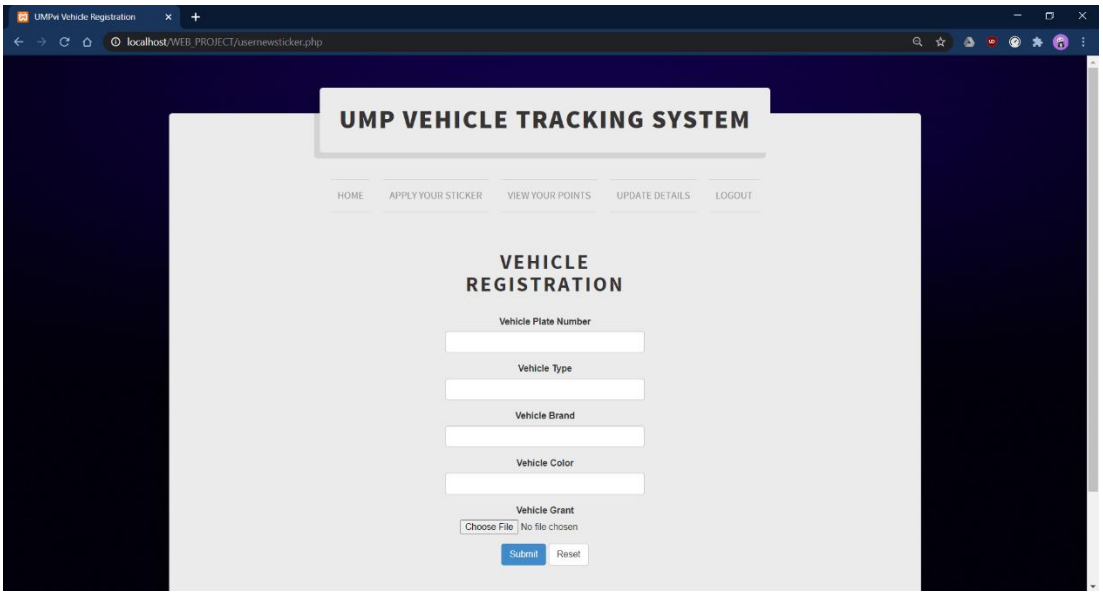


Figure 9: Homepage (User) Interface

Then for this module, the user needs to apply their vehicle sticker to be pasted at their vehicle. So, they need to register their vehicle in the system where several information of their vehicle such as plate number, type, brand, and colour. The applicants also need to upload their vehicle grant document for approval by the administrator. The sample of registration form is shown below.



Admin Interface: Approving application

Upon vehicle registration process is complete, the user will be redirected to the next module which is the generation of the vehicle sticker. But before that, the administrator needs to approve the application first before the user can get the new sticker. The admin panel to approve those applications is shown below:

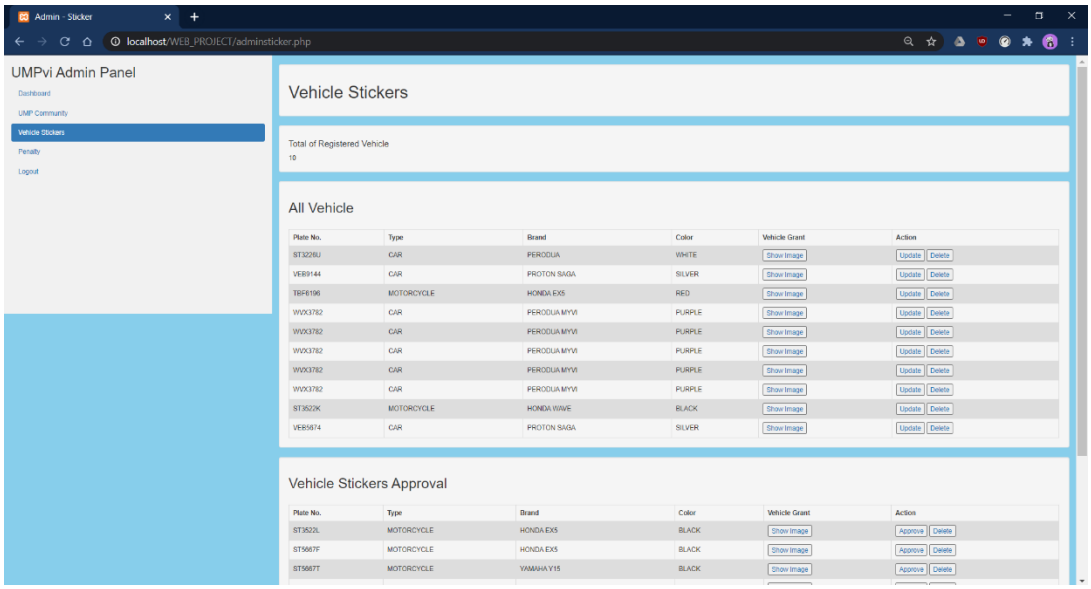


Figure 11: Sticker approval interface (Admin)

Module 2: Manage QR Codes

In the user page, continuing the vehicle registration page, is the generation of vehicle sticker together with the unique QR codes embedded with it. To generate this sticker, a One-Time Passcode (OTP) is required for the user to enter their Matric ID and Vehicle Plate in order to validate their applications. Then the sticker will be generated but as a view-only mode as the user need to go to the UMP Safety Department (Bahagian Keselamatan) to get their sticker application approved and printed.

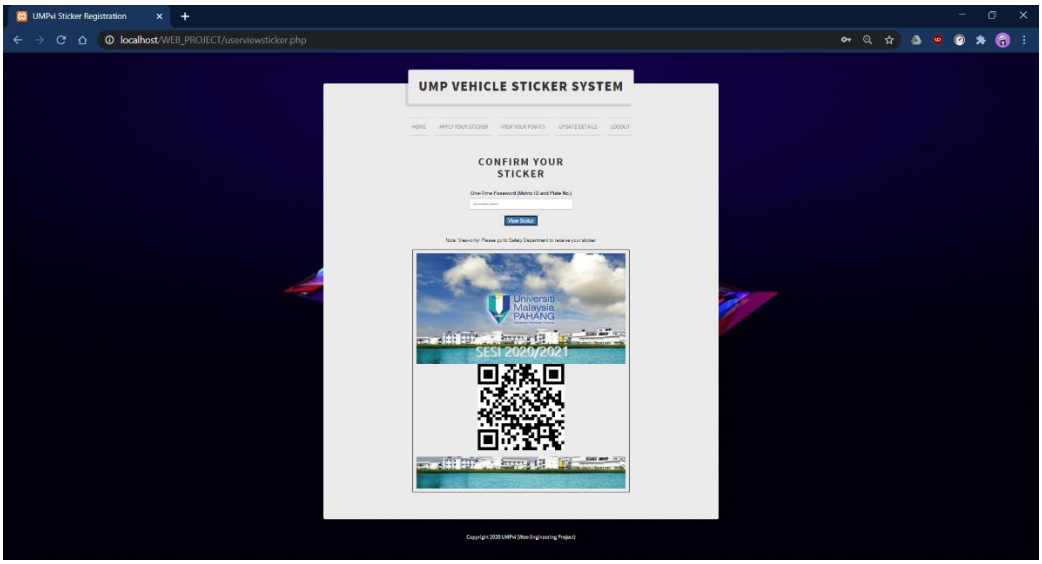


Figure 12: Sticker with QR code generation

The administrator needs to print the physical sticker that have been approved to be given to the user. The user needs to paste their vehicle sticker and shown in their vehicle to grant access to enter UMP area as well as the tag for the UMP Safety Department officer to detect or issuing any penalty later.

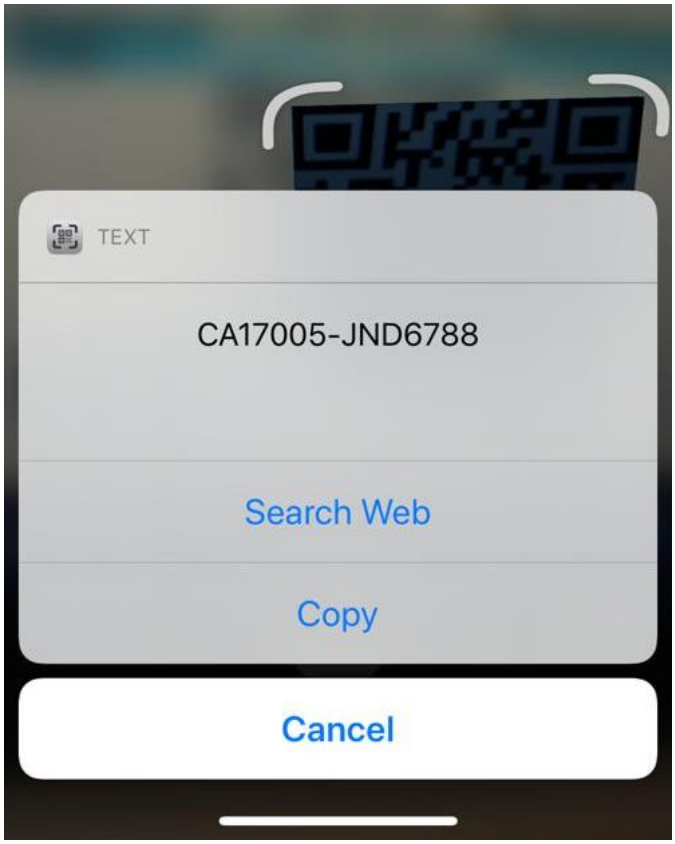


Figure 13: QR Code scanned on the sticker

The Figure 13 above shows the scanned QR Code on the sticker to provide the safety officer to take note on the details for the vehicle that they are checking. This is to ease them to issue penalties to the vehicle that disobeying the UMP traffic rules as the owner matric ID and the vehicle plate number is shown when the sticker is scanned. This is also helpful in detecting whether a right vehicle is using the right sticker as nowadays the case of illegal sticker duplication is increasing each day.

Module 3: Manage Traffic Violation

Continuing the 2nd module, is the Manage Traffic Violation module which is specifically used only by the administrator where it helps them in issuing penalty. The interface will require several informations of the penalty that needs to be inserted by the safety officer such as the penalty time, date, and also the type of violation. The officer also need to enter their credentials as administrator in order to validate the penalty issuing. If the system detects that the input is not on the admin database, it will shows an error and the penalty would not be issued. The figure below shows the interface of the Issue Penalty:

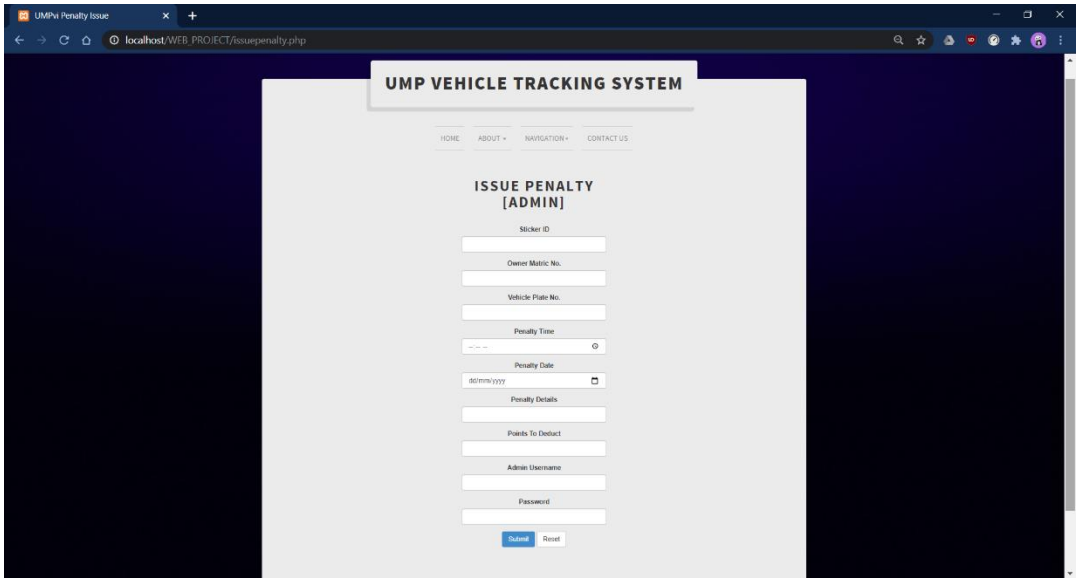


Figure 14: Issue Penalty interface

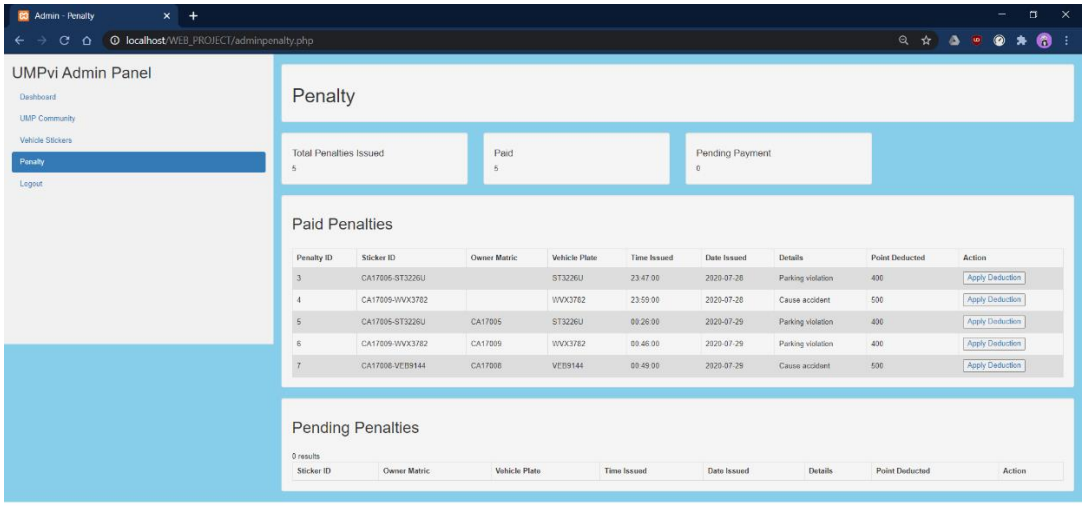


Figure 15: Penalty that have been issued (Admin panel)

Module 4: Manage Points

When a penalty is issued, the user yearly point collected will be deducted depending on the type of violation that is done by the vehicle owner. So, the admin can deduct the user points by accessing the admin panel at the Penalty tab. The user will be required to pay for the penalty that they have been issued. Before their points are deducted, the status of the penalty will be on pending and only will be settled after they have paid the penalty and the admin will apply deduction on the admin panel. The figure below shows on the penalty lists:

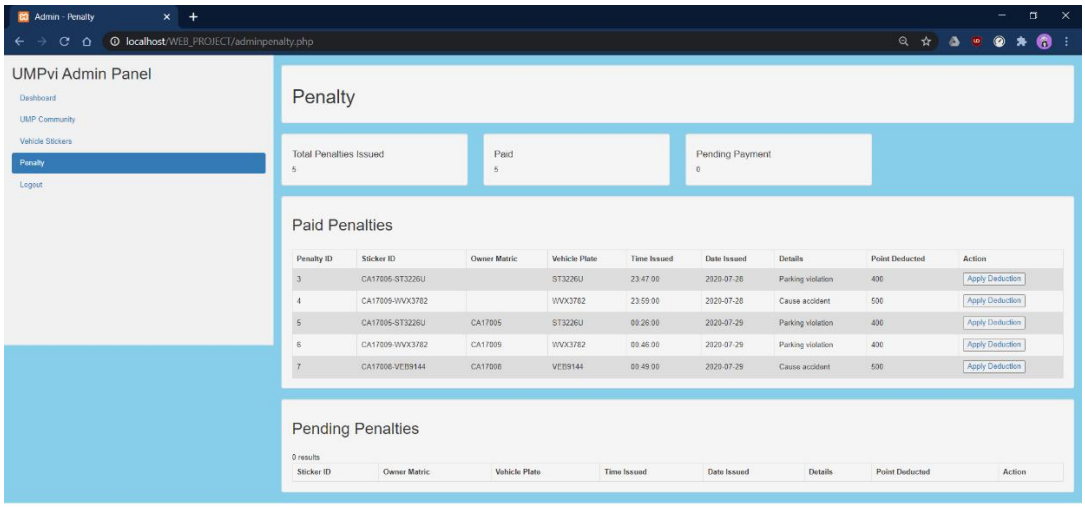


Figure 16: The list of paid penalties

For the user, they can check and review their current points where it is updated automatically after any deduction or addition of points. The sample of the 'View Point' interface is shown below:

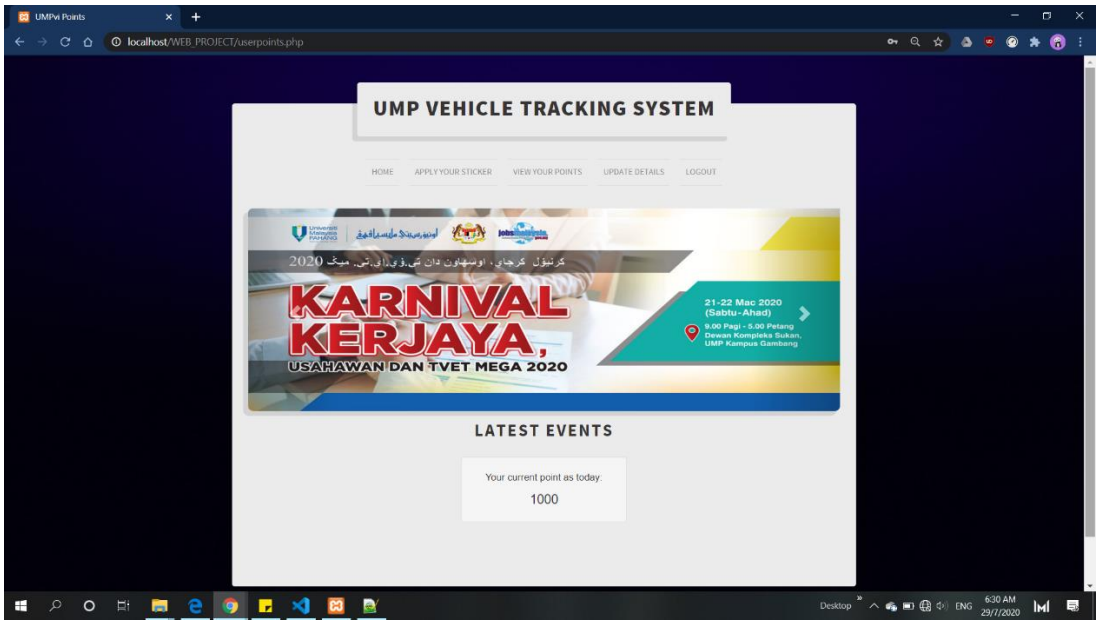


Figure 17: View point interface (User)

Module 6: Manage Reports, Data Analytics & Login

This module is used by the administrator only, where they can observe the figure counter directly on their admin panel such as the total number of user (UMP Community), total number of registered vehicles as well as the total penalty that have been issued. The admin also can generate an instant report that will be downloaded as an Excel file for printing and documentation purpose as shown in the figure below:

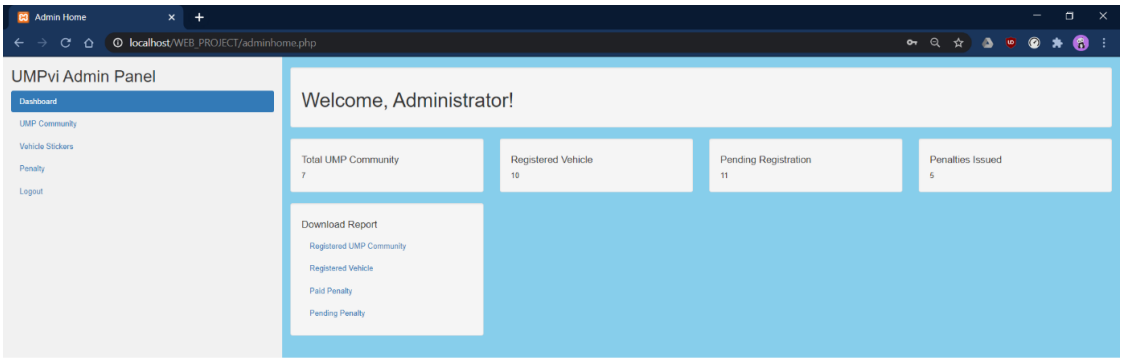


Figure 18: Admin panel

Lastly, Module 6 also will handle the login page. There are in total 3 link in the login page. First is the login page for the user (UMP Community). Second is for the admin panel (Admin) and lastly is for the issuing of penalty by the officer.

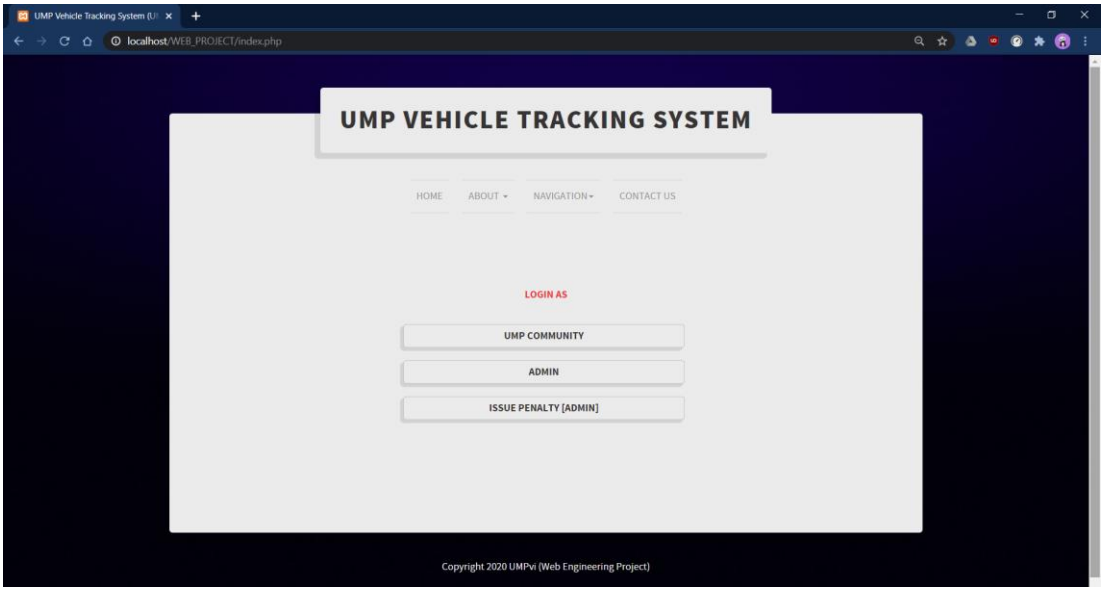


Figure 19: Login page

SQL CODE

```
-- phpMyAdmin SQL Dump
-- version 4.9.2
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: Jul 29, 2020 at 01:51 AM
-- Server version: 10.4.10-MariaDB
-- PHP Version: 7.3.12

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET AUTOCOMMIT = 0;
START TRANSACTION;
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `umpvi`
--

-- --------------------------------------------------------

--
-- Table structure for table `admin`
--

CREATE TABLE `admin` (
  `ADMIN_ID` int(11) NOT NULL,
  `USERNAME` varchar(100) NOT NULL,
  `PASSWORD` varchar(20) NOT NULL
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `admin`
--

INSERT INTO `admin` (`ADMIN_ID`, `USERNAME`, `PASSWORD`) VALUES
(1, 'AZAM', 'azam123');

-- --------------------------------------------------------

--
-- Table structure for table `fileupload`
--

CREATE TABLE `fileupload` (
  `ID` int(11) NOT NULL,
  `FILENAME` varchar(255) CHARACTER SET utf8 COLLATE utf8_unicode_ci NOT NULL,
  `UPLOADEDON` datetime NOT NULL,
  `STATUS` enum('0','1') NOT NULL DEFAULT '1'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Table structure for table `penalty`
--

CREATE TABLE `penalty` (
  `PENALTY_ID` int(11) NOT NULL,
  `STICKER_ID` varchar(20) NOT NULL,
  `USER_MATRIC` varchar(20) NOT NULL,
  `VEHICLE_PLATE` varchar(100) NOT NULL,
  `PENALTY_TIME` time NOT NULL,
  `PENALTY_DATE` date NOT NULL,
```

```
  `PENALTY_DETAILS` varchar(200) NOT NULL,
  `POINT_ID` int(11) NOT NULL,
  `PENALTY_STATUS` enum('0','1') NOT NULL DEFAULT '0'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `penalty`
--

INSERT INTO `penalty` (`PENALTY_ID`, `STICKER_ID`, `USER_MATRIC`, `VEHICLE_PLATE`, `PENALTY_TIME`, `PENALTY_DATE`, `PENALTY_DETAILS`, `POINT_ID`, `PENALTY_STATUS`) VALUES
(3, 'CA17005-ST3226U', '', 'ST3226U', '23:47:00', '2020-07-28', 'Parking violation', 400, '1'),
(4, 'CA17009-WVX3782', '', 'WVX3782', '23:59:00', '2020-07-28', 'Cause accident', 500, '1'),
(5, 'CA17005-ST3226U', 'CA17005', 'ST3226U', '00:26:00', '2020-07-29', 'Parking violation', 400, '1'),
(6, 'CA17009-WVX3782', 'CA17009', 'WVX3782', '00:46:00', '2020-07-29', 'Parking violation', 400, '1'),
(7, 'CA17008-VEB9144', 'CA17008', 'VEB9144', '00:49:00', '2020-07-29', 'Cause accident', 500, '1');

-- --------------------------------------------------------

--
-- Table structure for table `points`
--

CREATE TABLE `points` (
  `POINT_ID` int(11) NOT NULL,
  `POINT_TOTAL` int(11) NOT NULL,
  `POINT_DETAIL` varchar(100) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `points`
--

INSERT INTO `points` (`POINT_ID`, `POINT_TOTAL`, `POINT_DETAIL`) VALUES
(1, 500, 'Cause Accident'),
```

```
(2, 400, 'Parking violation'),
(3, 300, 'No sticker displayed'),
(4, 200, 'Not wearing seat belt or helmet');

-- --------------------------------------------------------

--
-- Table structure for table `sticker`
--

CREATE TABLE `sticker` (
  `STICKER_ID` int(11) NOT NULL,
  `USER_NAME` varchar(50) NOT NULL,
  `USER_MATRIC` varchar(10) NOT NULL,
  `VEHICLE_PLATE` varchar(20) NOT NULL,
  `VEHICLE_TYPE` varchar(20) NOT NULL,
  `STATUS` enum('0','1') NOT NULL DEFAULT '0'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Table structure for table `user`
--

CREATE TABLE `user` (
  `USER_ID` int(11) NOT NULL,
  `USER_NAME` varchar(50) NOT NULL,
  `USER_MATRIC` varchar(10) NOT NULL,
  `USER_EMAIL` varchar(100) NOT NULL,
  `USER_PASSWORD` varchar(20) NOT NULL,
  `USER_PHONE` varchar(20) NOT NULL,
  `USER_DEPT` varchar(10) NOT NULL,
  `USER_POINT` int(11) NOT NULL DEFAULT 1000
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
--

-- Dumping data for table `user`

--


INSERT INTO `user` (`USER_ID`, `USER_NAME`, `USER_MATRIC`, `USER_EMAIL`, `USER_PASSWORD`, `USER_PHONE`, `USER_DEPT`, `USER_POINT`) VALUES

(2, 'MUHAMMAD HILMI BIN ALI', 'CA17009', 'javier.hilmi@gmail.com', 'hilmikurus', '0198876622', 'FKOM', 600),

(3, 'MUHAMMAD KHAIRI AFIFI BIN ABD RAZAK', 'CA17118', 'afifi_khairi@gmail.com', 'fifi456', '0123324455', 'FIST', 1000),

(4, 'MOHD RAFI RIFFLI BIN RAFIZAL', 'CA17023', 'rafi@gmail.com', 'rafi789', '0134456644', 'FKKSA', 1000),

(6, 'AHMAD ZAKI BIN HAMADI', 'KA16732', 'superzaki@facebook.com', 'zaki123', '0123323322', 'FTEK', 1000),

(7, 'MUHAMMAD TALHAH BIN MOHAMED NASIR', 'CA17008', 'stole14@gmail.com', 'stole14', '0187652322', 'FIM', 500),

(9, 'MUHAMMAD SYAFIQ BIN ABAS', 'CA17121', 'syafiq.abas@gmail.com', 'syafiq789', '0197886655', 'UMPH', 1000),

(10, 'MOHD ELFARIZ BIN MOHD IRFAN', 'CA17005', 'elfariz@yahoo.com.my', 'elfariz123', '0109351963', 'FKOM', 1000);


-- --------------------------------------------------------


--

-- Table structure for table `vehicle`

--


CREATE TABLE `vehicle` (

  `VEHICLE_ID` int(11) NOT NULL,

  `VEHICLE_PLATE` varchar(100) NOT NULL,

  `VEHICLE_TYPE` varchar(20) NOT NULL,

  `VEHICLE_BRAND` varchar(100) NOT NULL,

  `VEHICLE_COLOR` varchar(100) NOT NULL,

  `VEHICLE_STATUS` enum('0','1') NOT NULL DEFAULT '0'

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;


--

-- Dumping data for table `vehicle`
```

```
--

INSERT INTO `vehicle` (`VEHICLE_ID`, `VEHICLE_PLATE`, `VEHICLE_TYPE`, `VEHICLE_BRAND`, `VEHICLE_COLOR`, `VEHICLE_STATUS`) VALUES
(2, 'ST3226U', 'CAR', 'PERODUA', 'WHITE', '1'),
(3, 'VEB9144', 'CAR', 'PROTON SAGA', 'SILVER', '1'),
(4, 'TBF6196', 'MOTORCYCLE', 'HONDA EX5', 'RED', '1'),
(5, 'WVX3782', 'CAR', 'PERODUA MYVI', 'PURPLE', '1'),
(6, 'WVX3782', 'CAR', 'PERODUA MYVI', 'PURPLE', '1'),
(7, 'WVX3782', 'CAR', 'PERODUA MYVI', 'PURPLE', '1'),
(8, 'WVX3782', 'CAR', 'PERODUA MYVI', 'PURPLE', '1'),
(9, 'WVX3782', 'CAR', 'PERODUA MYVI', 'PURPLE', '1'),
(10, 'ST3522K', 'MOTORCYCLE', 'HONDA WAVE', 'BLACK', '1'),
(11, 'ST3522L', 'MOTORCYCLE', 'HONDA EX5', 'BLACK', '0'),
(12, 'ST5667F', 'MOTORCYCLE', 'HONDA EX5', 'BLACK', '0'),
(13, 'ST5667T', 'MOTORCYCLE', 'YAMAHA Y15', 'BLACK', '0'),
(14, 'ST5667U', 'MOTORCYCLE', 'YAMAHA Y15', 'BLACK', '0'),
(15, 'TAG6627', 'CAR', 'PERODUA KELISA', 'ORANGE', '0'),
(16, 'ST5154L', 'MOTORCYCLE', 'HONDA EX5', 'BLUE', '0'),
(17, 'JND7835', 'CAR', 'PERODUA MYVI', 'YELLOW', '0'),
(18, 'ST7456R', 'CAR', 'TOYOTA AVANZA', 'WHITE', '0'),
(19, 'ST5667A', 'MOTORCYCLE', 'HONDA EX5', 'PURPLE', '0'),
(20, 'VEB5674', 'CAR', 'PROTON SAGA', 'SILVER', '1'),
(21, 'BEA6677', 'MOTORCYCLE', 'YAMAHA Y15', 'YELLOW', '0'),
(22, 'WQH7000', 'CAR', 'PERODUA VIVA', 'SILVER', '0');

--
-- Indexes for dumped tables
--

--
-- Indexes for table `admin`
--
ALTER TABLE `admin`
  ADD PRIMARY KEY (`ADMIN_ID`);
```

```
--
-- Indexes for table `fileupload`
--
ALTER TABLE `fileupload`
  ADD PRIMARY KEY (`ID`);

--
-- Indexes for table `penalty`
--
ALTER TABLE `penalty`
  ADD PRIMARY KEY (`PENALTY_ID`);

--
-- Indexes for table `points`
--
ALTER TABLE `points`
  ADD PRIMARY KEY (`POINT_ID`);

--
-- Indexes for table `sticker`
--
ALTER TABLE `sticker`
  ADD PRIMARY KEY (`STICKER_ID`);

--
-- Indexes for table `user`
--
ALTER TABLE `user`
  ADD PRIMARY KEY (`USER_ID`);

--
-- Indexes for table `vehicle`
--
ALTER TABLE `vehicle`
  ADD PRIMARY KEY (`VEHICLE_ID`);
```

```
--
-- AUTO_INCREMENT for dumped tables
--


--
-- AUTO_INCREMENT for table `admin`
--
ALTER TABLE `admin`
  MODIFY `ADMIN_ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;


--
-- AUTO_INCREMENT for table `fileupload`
--
ALTER TABLE `fileupload`
  MODIFY `ID` int(11) NOT NULL AUTO_INCREMENT;


--
-- AUTO_INCREMENT for table `penalty`
--
ALTER TABLE `penalty`
  MODIFY `PENALTY_ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=8;


--
-- AUTO_INCREMENT for table `points`
--
ALTER TABLE `points`
  MODIFY `POINT_ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;


--
-- AUTO_INCREMENT for table `sticker`
--
ALTER TABLE `sticker`
  MODIFY `STICKER_ID` int(11) NOT NULL AUTO_INCREMENT;
```

```
--
-- AUTO_INCREMENT for table `user`
--
ALTER TABLE `user`
  MODIFY `USER_ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;

--
-- AUTO_INCREMENT for table `vehicle`
--
ALTER TABLE `vehicle`
  MODIFY `VEHICLE_ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=23;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```