Wireless Sensor-based Application for Managing Theme Park

Software Design Specification

**By**

Pabhawee Chuacharoen 552115037

Department of Software Engineering

College of Arts, Media and Technology

Chiang Mai University

Project Advisor

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr.Noppon Choosri

**Document History**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Version | Status | Change Context | Viewable | Reviewer | Responsible | Date |
| V.1.0 | Draft | Introduction  System Architecture | PC, NC | PC | PC | 12/08/15 |
| V.1.1 | Draft | Class Description  ER Diagram | PC, NC | PC | PC | 13/08/15 |
| V.1.2 | Draft | Class Description  Sequence Diagram | PC, NC | PC, NC | PC | 16/08/15 |
| V.2.0 | Release | Change project name  Fix Sequence diagram | PC, NC | PC, NC | PC | 27/08/15 |

PC = Pabhawee Chuacharoen , NC = Dr. Noppon Choosri

# Table of Content

[Table of Content 2](#_Toc434401203)

[Chapter One | Introduction 4](#_Toc434401204)

[1.1. Purpose 4](#_Toc434401205)

[1.2. Software Scope 4](#_Toc434401206)

[1.3. Acronyms and Definitions 4](#_Toc434401207)

[1.3.1. Key Acronyms and Abbreviation 4](#_Toc434401208)

[Chapter Two | System Architecture 5](#_Toc434401209)

[Chapter Three | Detailed Design 6](#_Toc434401210)

[3.1. Class Diagram Description 6](#_Toc434401211)

[3.1.1. Wireless sensor-based application System overall class diagram 6](#_Toc434401212)

[Chapter Four | Class Description 9](#_Toc434401213)

[4.1. Class-01: Attraction 9](#_Toc434401214)

[4.2. Class-02: Guest 10](#_Toc434401215)

[4.3. Class-03: RegisterData 11](#_Toc434401216)

[4.4. Class-04: Attractions 11](#_Toc434401217)

[4.5. Class-05: Guests 12](#_Toc434401218)

[4.6. Class-06: Postdata 13](#_Toc434401219)

[4.7. Class-07: attraction\_form 14](#_Toc434401220)

[4.8. Class-08: attraction\_edit\_form 14](#_Toc434401221)

[4.9. Class-09: attraction\_view 14](#_Toc434401222)

[4.10. Class-10: guest\_form 14](#_Toc434401223)

[4.11. Class-11: guest\_edit\_form 15](#_Toc434401224)

[4.12. Class-12: guest\_view 15](#_Toc434401225)

[4.13. Class-13: ActivityStat 15](#_Toc434401226)

[4.14. Class-14: Activity 16](#_Toc434401227)

[4.15. Class-15: activity\_view 16](#_Toc434401228)

[Chapter Five | Sequence Diagram 17](#_Toc434401229)

[5.1 SD-01: URS-01 Ticket seller can register guest into the system. 17](#_Toc434401230)

[5.2 SD-02: URS-02 Ticket seller can view guest information from the system 18](#_Toc434401231)

[5.3 SD-03: URS-03 Ticket seller can edit guest information from the system. 19](#_Toc434401232)

[5.4 SD-04: URS-04 Ticket seller can delete guest information from the system. 20](#_Toc434401233)

[5.5 SD-05: to URS-05 Admin can view attraction information from the system. 21](#_Toc434401234)

[5.6 SD-06: URS-06 Admin can add attraction information into the system. 22](#_Toc434401235)

[5.7 SD-07: URS-07 Admin can edit attraction information from the system. 23](#_Toc434401236)

[5.8 SD-08: URS-18 Admin can delete attraction information from the system. 24](#_Toc434401237)

[5.9 SD-09: URS-09: Guest can enter the entrance gate by representing RFID tag at RFID reader. 25](#_Toc434401238)

[5.10 SD-10: URS-10: Guest can exit the exit gate by representing RFID tag at RFID reader. 25](#_Toc434401239)

[5.11 SD-11: URS-11: Guest can get through attraction entrance by representing RFID tag at RFID reader. 26](#_Toc434401240)

[5.12 SD-12: Admin can view a report on numbers of guest visiting the theme park within one week range. 27](#_Toc434401241)

[5.13 SD-13: URS-13: Admin can view a report on numbers of guest attending each ride within one week range. 28](#_Toc434401242)

[Chapter Six | ER Diagram 29](#_Toc434401243)

[Chapter Seven | UI Design 30](#_Toc434401244)

[UI-01: Home page 30](#_Toc434401245)

[UI-02: Guest register form 30](#_Toc434401246)

[UI-03: Guest list 31](#_Toc434401247)

[UI-04: Guest edit form 31](#_Toc434401248)

[UI-05: Attraction form 32](#_Toc434401249)

[UI-06: Attraction list 32](#_Toc434401250)

[UI-07: Attraction edit form 33](#_Toc434401251)

[UI-08: Activity line chart 33](#_Toc434401252)

[Chapter Eight | Wiring Schemes 34](#_Toc434401253)

[8.1 Pinout 34](#_Toc434401254)

# Chapter One | Introduction

## Purpose

The software design document (SDD) is written for describing the high level of software design, system architecture, database system and graphic user interfaces. Also consists of the users’ requirement. This SDD is the baseline for limited changing in the scope of the system. The important thing, this document is the blueprint for facilitating the developer team members to understand the direction of the system.

## Software Scope

This document describes the implementation details of Wireless sensor-based application for managing theme park

The main features of Wireless sensor-based application for managing theme park Project will be as follows:

* **Guest information management**

This feature supports ticket seller and admin in managing customer information in the system, it covers most of the guest information manipulation such as register/edit/delete guest information.

* **Attraction information management**

This feature supports Admin in managing attraction information in the system; it covers most of the attraction information manipulation such as add/edit/delete attraction information.

## Acronyms and Definitions

### Key Acronyms and Abbreviation

|  |  |
| --- | --- |
| Name | Explain |
| WSAT | Wireless sensor-based application for managing theme park |
| GIMS | Guest information management system |
| AIMS | Attraction information management system |
| SRS | Software Requirement Specification |
| URS | User Requirement Specification |
| UC | Use Case Description |
| UI | User Interface |
| UID | User Identification |
| RFID | Radio-frequency identification |
| ER | Entity–relationship |

# Chapter Two | System Architecture

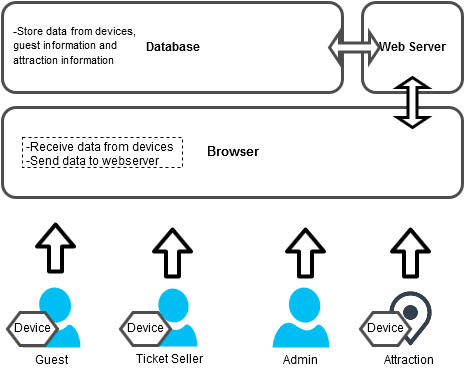


Figure 1: System architecture

The architecture of Theme Park Management System by using RFID Technology , shown in figure-1, consists of several parts:

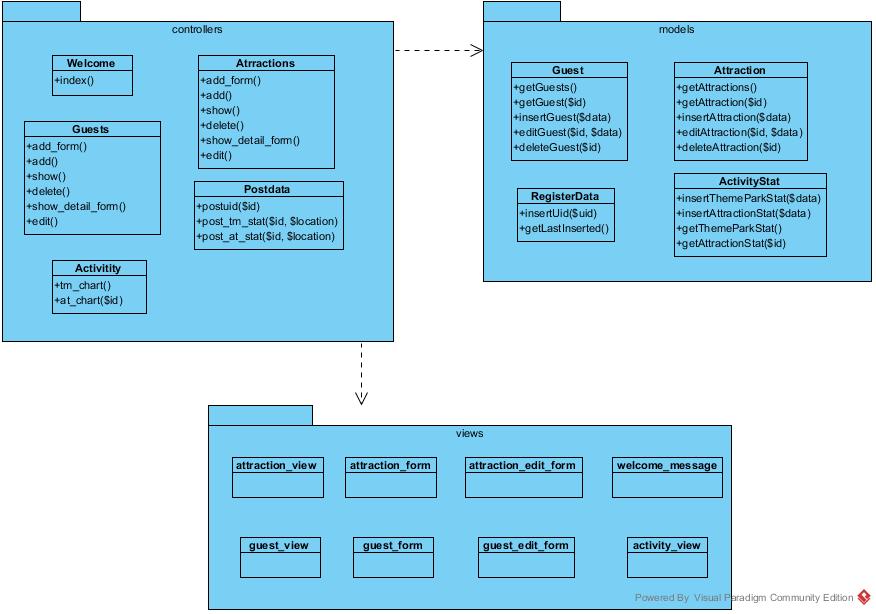
* **Arduino:** The arduino connects to RFID module and Wi-Fi module. It’ll be deployed alongside with stations, entrance gate, exit gate rides’ entrance and ticket seller boot. The main task is to send data from arduino to web server. The person who will use this are guest and ticket seller.
* **Web browser :**The application that display the data, which come from Web Service.
* **Web server :**The container that store web data and provide web service.
* **Database :**The internal database of Theme Park Management System.

# Chapter Three | Detailed Design

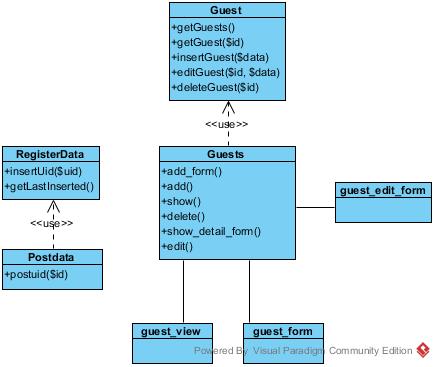
## Class Diagram Description

This section shows classes that support the features described in SRS. It describes the relationships between the classes in diagram. More detailed information of classes will be described in the Class Description section.

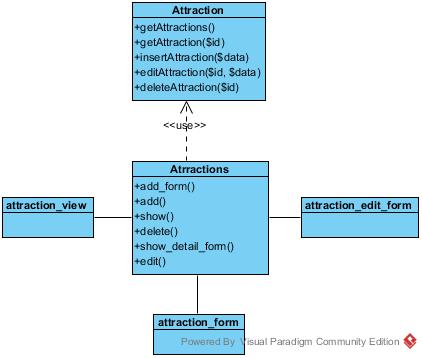
### Wireless sensor-based application System overall class diagram



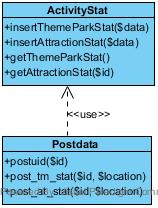
#### Classes related with Feature#1:Guest information management



#### Classes related with Feature#2: Attraction information management



#### Classes related with Feature#3: Collecting visiting statistics data for attraction information



#### Classes related with Feature#4: Attractions' statistic information

#### D:\UML\workspace\TPM\Act9-13\F4.jpg

# Chapter Four | Class Description

## Class-01: Attraction



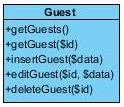
* **Description:**

This class contains function that gets, insert, edit and delete for attraction information.

* **Attributes: -**
* **Methods:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Method name | description | Parameter Lists | return type |
| 1 | **getAttractions()** | **This method is to get all attractions within the database** |  | **array** |
| 2 | **getAttraction(id)** | **This method is to get an attraction that has the same id as the parameter** | **$id** | **Object** |
| 3 | **insertAttraction(data)** | **This method is to insert an attraction into the database** | **$data** | **Object** |
| 4. | **editAttraction(id, data)** | **This method is to update an attraction that has the same id as the parameter** | **$id** | **Object** |
| 5 | **deleteAttraction(id)** | **This method is to get an attraction that has the same id as the parameter** | **$id** | **object** |

## Class-02: Guest



**Description:** This class contains function that get, insert, edit, delete and change the status for guest information.

* **Attributes: -**
* **Methods:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Method name | description | Parameter Lists | return type |
| 1 | **getGuests()** | **This method is to get all guests within the database** |  | **array** |
| 2 | **getGuest($id)** | **This method is to get a guest that has the same id as the parameter** | **$id** | **object** |
| 3 | **insertGuest($data)** | **This method is to insert a guest into the database** | **$data** | **object** |
| 4. | **editGuest($id,$ data)** | **This method is to update a guest that has the same id as the parameter** | **$id** | **object** |
| 5 | **deleteGuest($id)** | **This method is to get a guest that has the same id as the parameter** | **$id** | **object** |

## 

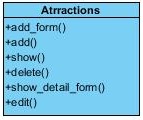
## Class-03: RegisterData

C:\Users\ASUS\Documents\VPProjects\registerdata.png

* **Description:**This class contains method that support guest registration by adding and getting data in the database.
* **Attributes: -**
* **Methods:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Method name | description | Parameter Lists | return type |
| 1 | **insertUid($uid)** | **This method is to insert an uid into the database** | **$uid** | **String** |
| 2 | **getLastInserted()** | **This method is to get the last inserted uid in the database from registerdata table** |  | **object** |

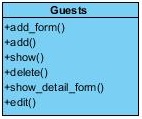
## Class-04: Attractions



* **Description:** This class is used to add, show, edit and delete an attraction by receiving/ sending data from/to other view classes and model classes.
* **Attributes: -**
* **Methods:**

| No. | Method name | description | Parameter Lists | return type |
| --- | --- | --- | --- | --- |
| 1 | **add\_form()** | **This method redirects users to attraction form** |  |  |
| 2 | **add()** | **This method receives input from view classes then call insertAttraction() method to add the data to the database** |  |  |
| 3 | **show()** | **This method is to get all attraction information by calling getAttractions() then pass it to attraction\_view in order to display it** |  |  |
| 4 | **delete()** | **This method is to delete an attraction by calling deleteAttraction from Attraction class then redirect the admin to the displayed page** |  |  |
| 5 | **Show\_detail\_form()** | **This method is to call getAttractions() then send the received data to display in guest\_edit\_form** |  |  |
| 6 | **edit()** | **This method is to update the received input by call editAttraction() method from Attraction lass** |  |  |

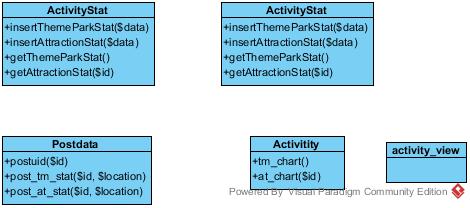
## Class-05: Guests



* **Description:** This class is used to add, show, edit and delete a guest by receiving/ sending data from/to other view classes and model classes.
* **Attributes: -**
* **Methods:**

| No. | Method name | description | Parameter Lists | return type |
| --- | --- | --- | --- | --- |
| 1 | **add\_form()** | **This method redirects users to guest registration form** |  |  |
| 2 | **add()** | **This receives input from view classes then call insertGuest() method to add the data to the database** |  |  |
| 3 | **show()** | **This method is to get all guest information by calling getGuests() then pass it to guest\_view in order to display it** |  |  |
| 4 | **delete()** | **This method is to delete a guest by calling deleteGuest () from Guest class then redirect the admin to the displayed page** |  |  |
| 5 | **show\_detail\_form()** | **This method is to call getGuests() then send the received data to display in guest\_edit\_form** |  |  |
| 6 | **edit()** | **This method is to call getGuests() then send the received data to display in attraction\_edit\_form** |  |  |

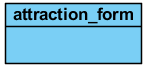
## Class-06: Postdata



* **Description:** This class is to handle the data that are sent by Arduino.
* **Attributes: -**
* **Methods:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Method name | description | Parameter Lists | return type |
| 1 | **Postid($id)** | **This method is to insert the received uid into the database in table name ‘registerdata** |  |  |
| 2 | **Post\_tm\_stat($id,$location)** | **This method is to insert the received uid and the location(i.e. entrance, exist gate) into the database in table name 'themepark\_attendance'** | **$id**  **$location** |  |
| 3 | **Post\_at\_stat($id,$location)** | **to insert the received uid and id of the attraction into the database in table name attraction\_attendance'** | **$id**  **$location** |  |

## Class-07: attraction\_form



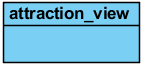
* **Description: This class is to provide input form and display error messages if any of the input is invalid.**
* **Attributes: -**
* **Methods: -**

## Class-08: attraction\_edit\_form



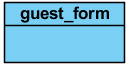
* **Description: This class is to provide an edit form and send the input to edit\_attraction for later operations.**
* **Attributes: -**
* **Methods: -**

## Class-09: attraction\_view



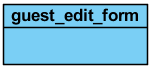
* **Description: This class is to display all attraction information in a table and provide edit and delete button for operations.**
* **Attributes: -**
* **Methods:**

## Class-10: guest\_form



* **Description: This class is to provide input form and display error messages if any of the input is invalid.**
* **Attributes: -**
* **Methods: -**

## Class-11: guest\_edit\_form



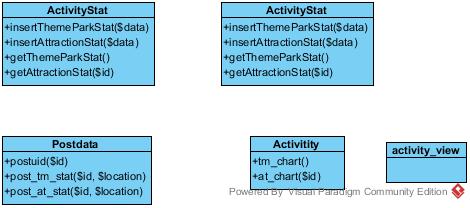
* **Description: This class is to provide an edit form and send the input to edit\_guest for later operations.**
* **Attributes: -**
* **Methods:-**

## Class-12: guest\_view



* **Description:: This class is to display all guest information in a table and provide edit and delete button for operations.**
* **Attributes: -**
* **Methods: -**

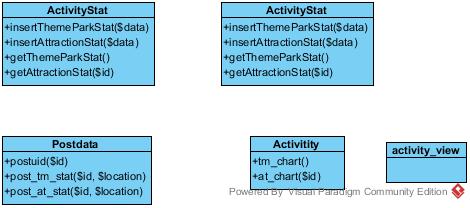
## Class-13: ActivityStat



* **Description:** This class is to handle the data that are sent by Arduino.
* **Attributes: -**
* **Methods:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Method name | description | Parameter Lists | return type |
| 1 | **insertThemeParkStat($data)** | **This method is to insert the received data from postdata class into the database in the 'tm\_stat' table** | **$data** |  |
| 2 | **insertAttractionStat($data)** | **This method is to insert the received data from postdata class into the database in the 'at\_stat' table** | **$data** |  |
| 3 | **getThemeParkStat()** | **This method is to get all the data that was inserted within range of one week count from the current date** |  | **Array Object** |
| 4 | **getAttractionStat($id)** | **This method is to get all the data that was inserted within range of one week count from the current date** | **$id** | **Array Object** |

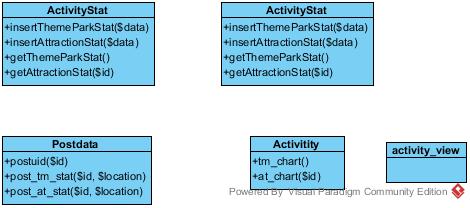
## Class-14: Activity



* **Description:** This class is to handle the data in order to populate line charts.
* **Attributes: -**
* **Methods:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Method name | description | Parameter Lists | return type |
| 1 | **tm\_chart()** | **This method is to request for the data the create a line chart to display numbers of visitor of the theme park from the said data** |  |  |
| 2 | **at\_chart()** | **This method is to request for the data the create a line chart to display numbers of visitor of the theme park from the said data** |  |  |

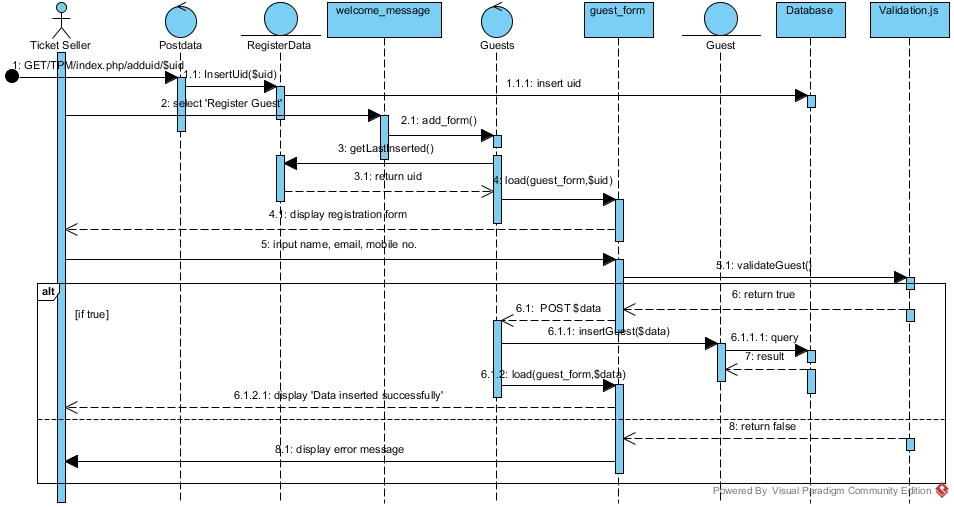
## Class-15: activity\_view



* **Description:: This class contains javascript of line chart to display all statistic information and a line chart of the theme park and attractions .**
* **Attributes: -**
* **Methods: -**

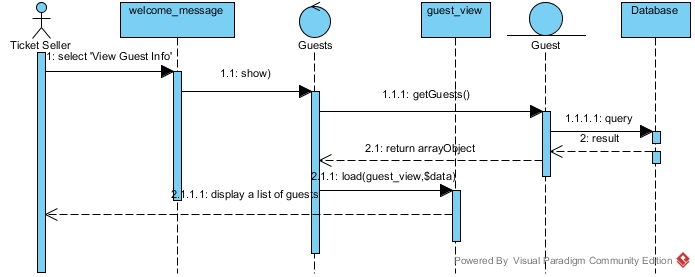
# Chapter Five | Sequence Diagram

## SD-01: URS-01 Ticket seller can register guest into the system.

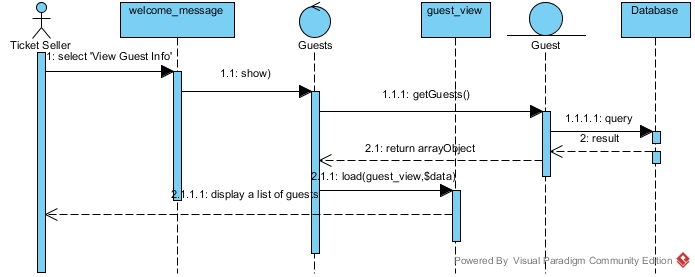


## 

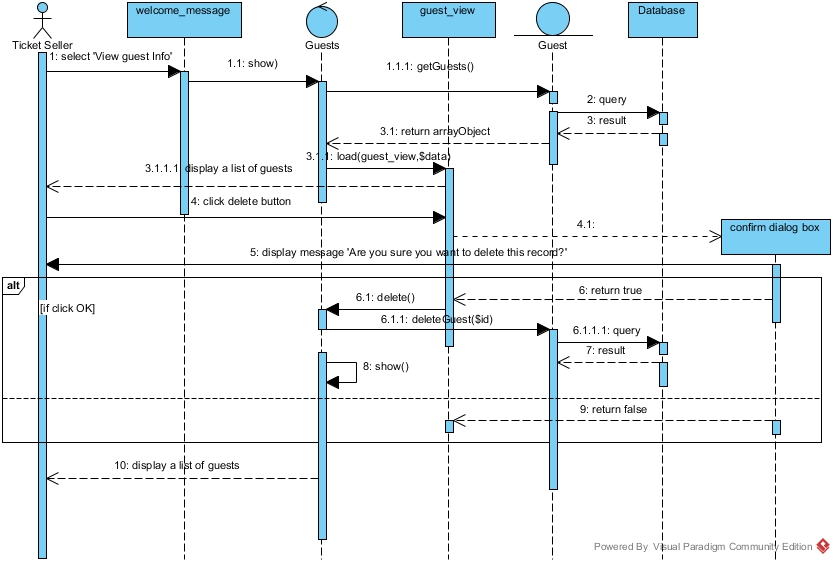
## SD-02: URS-02 Ticket seller can view guest information from the system



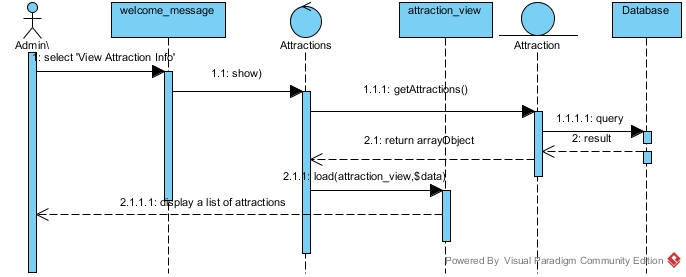
## SD-03: URS-03 Ticket seller can edit guest information from the system.



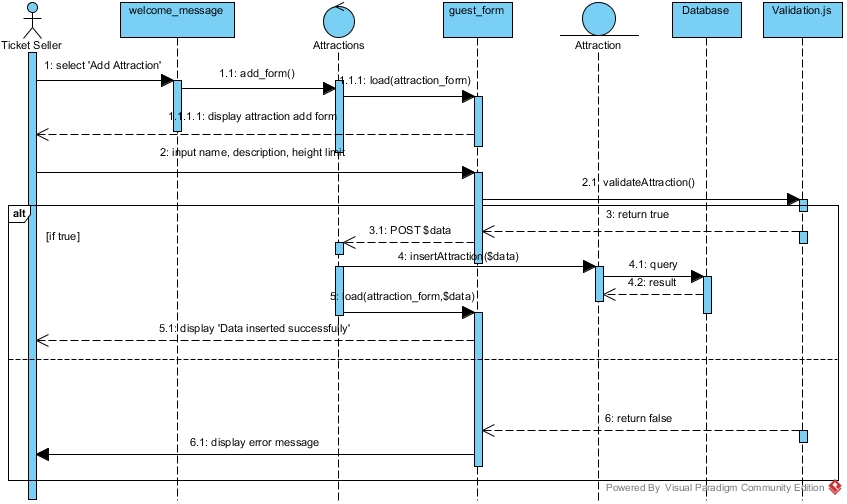
## SD-04: URS-04 Ticket seller can delete guest information from the system.



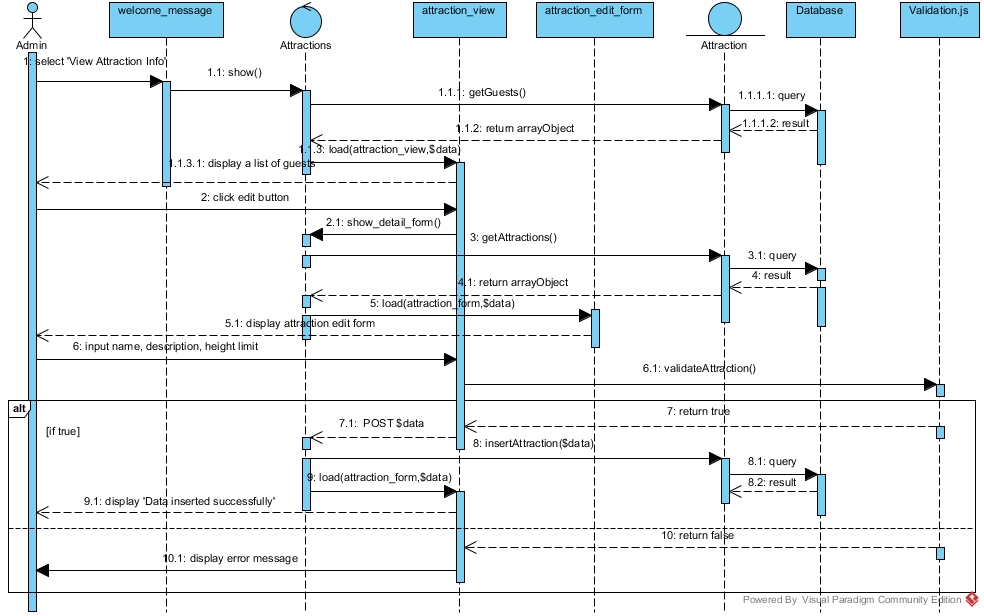
## SD-05: to URS-05 Admin can view attraction information from the system.



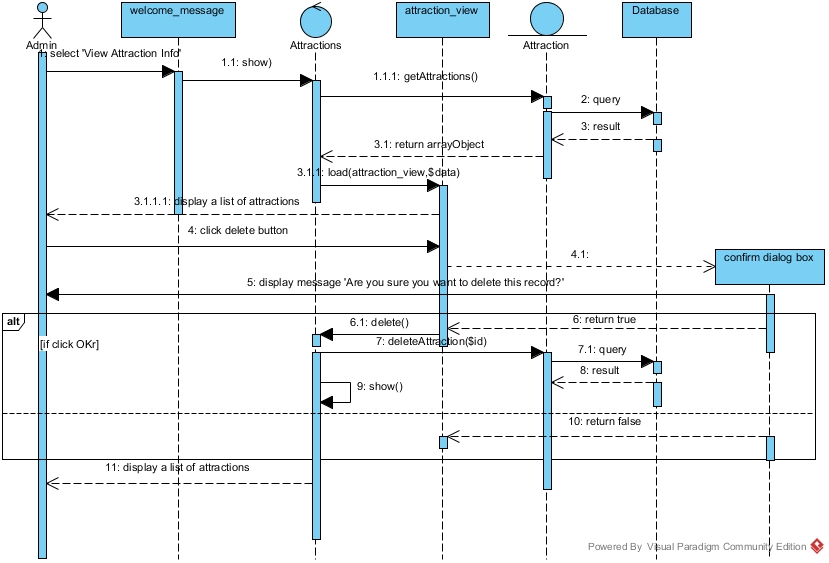
## SD-06: URS-06 Admin can add attraction information into the system.



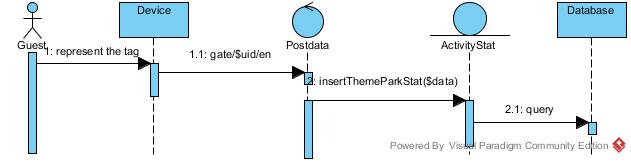
## SD-07: URS-07 Admin can edit attraction information from the system.



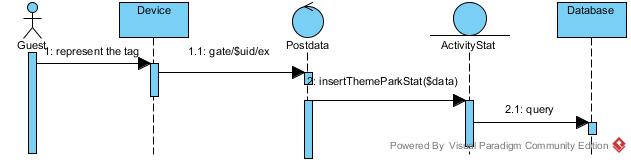
## SD-08: URS-18 Admin can delete attraction information from the system.



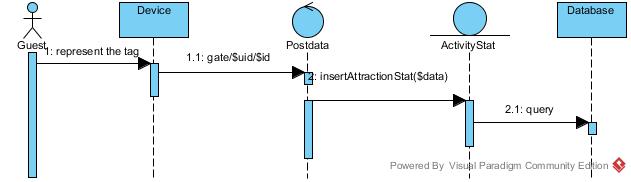
## SD-09: URS-09: Guest can enter the entrance gate by representing RFID tag at RFID reader.



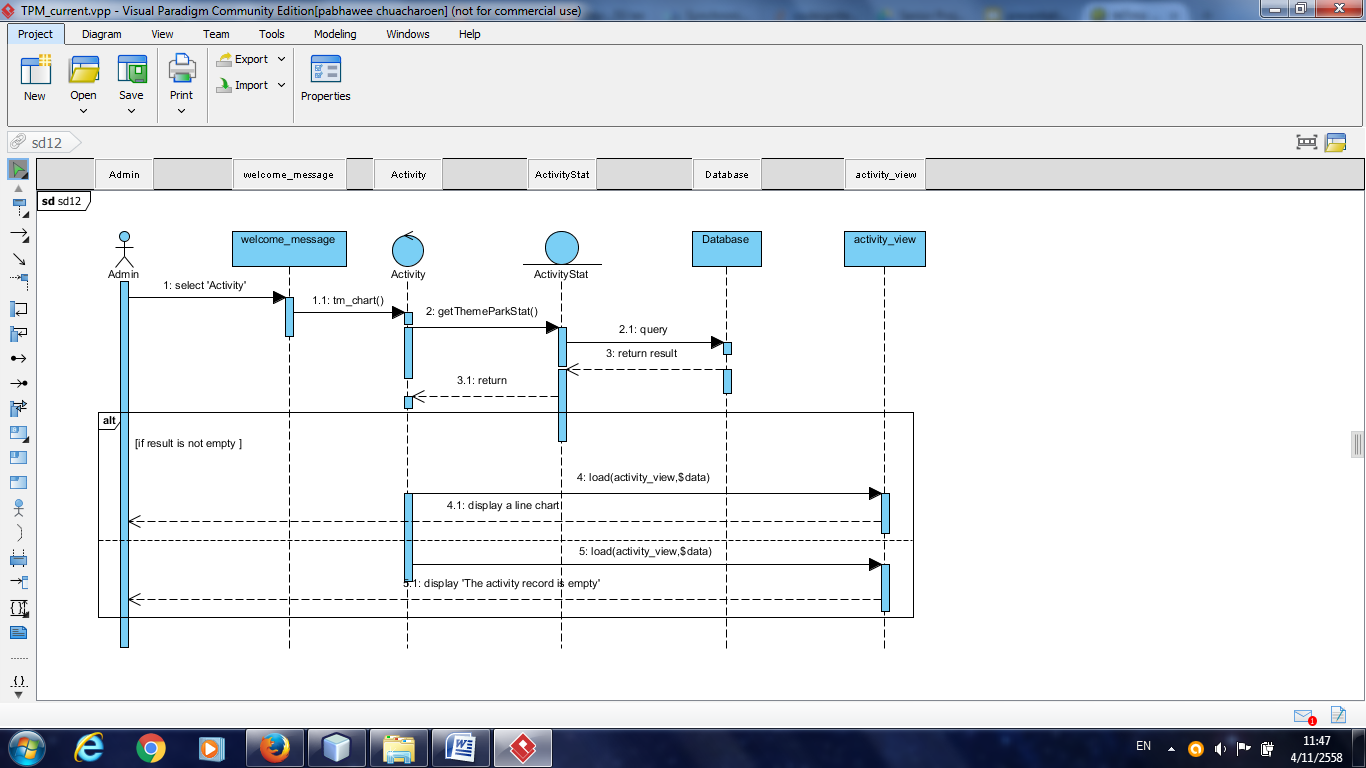
## SD-10: URS-10: Guest can exit the exit gate by representing RFID tag at RFID reader.



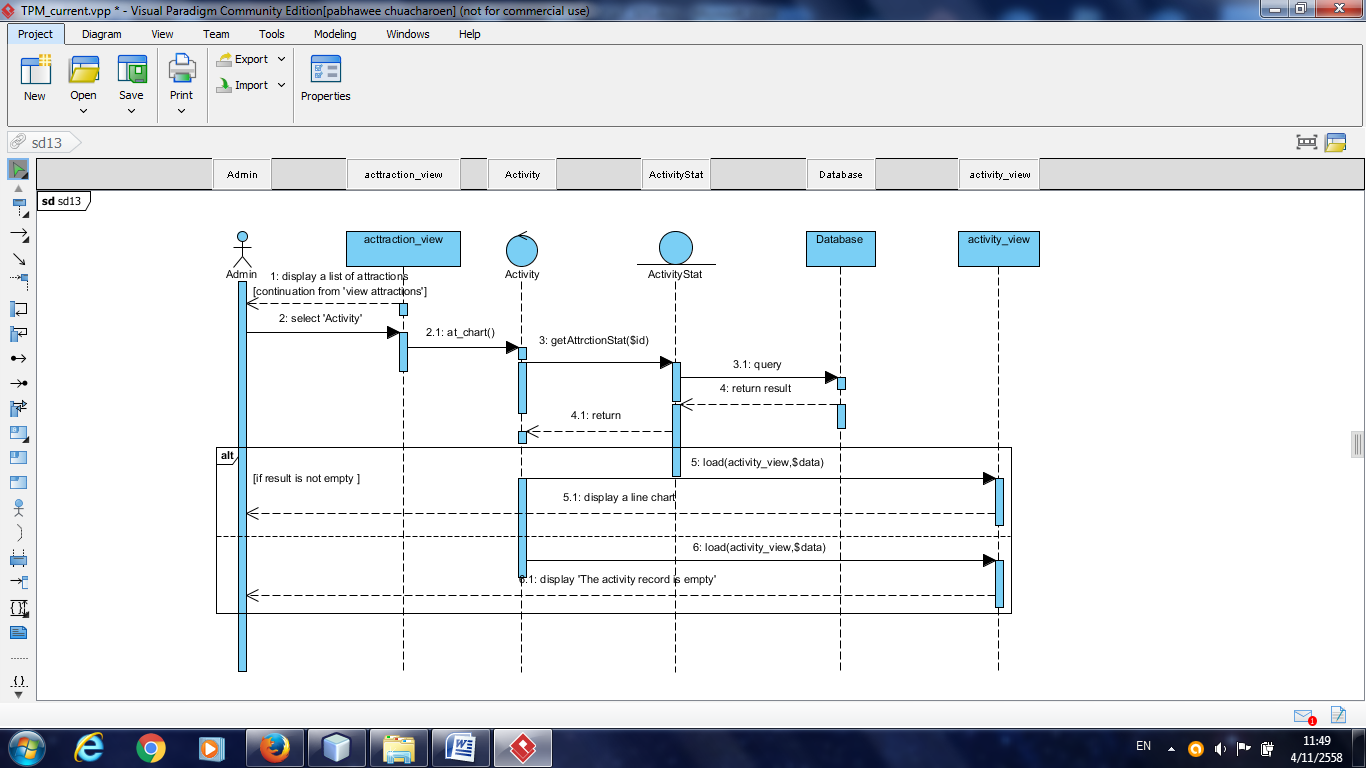
## SD-11: URS-11: Guest can get through attraction entrance by representing RFID tag at RFID reader.



## SD-12: Admin can view a report on numbers of guest visiting the theme park within one week range.

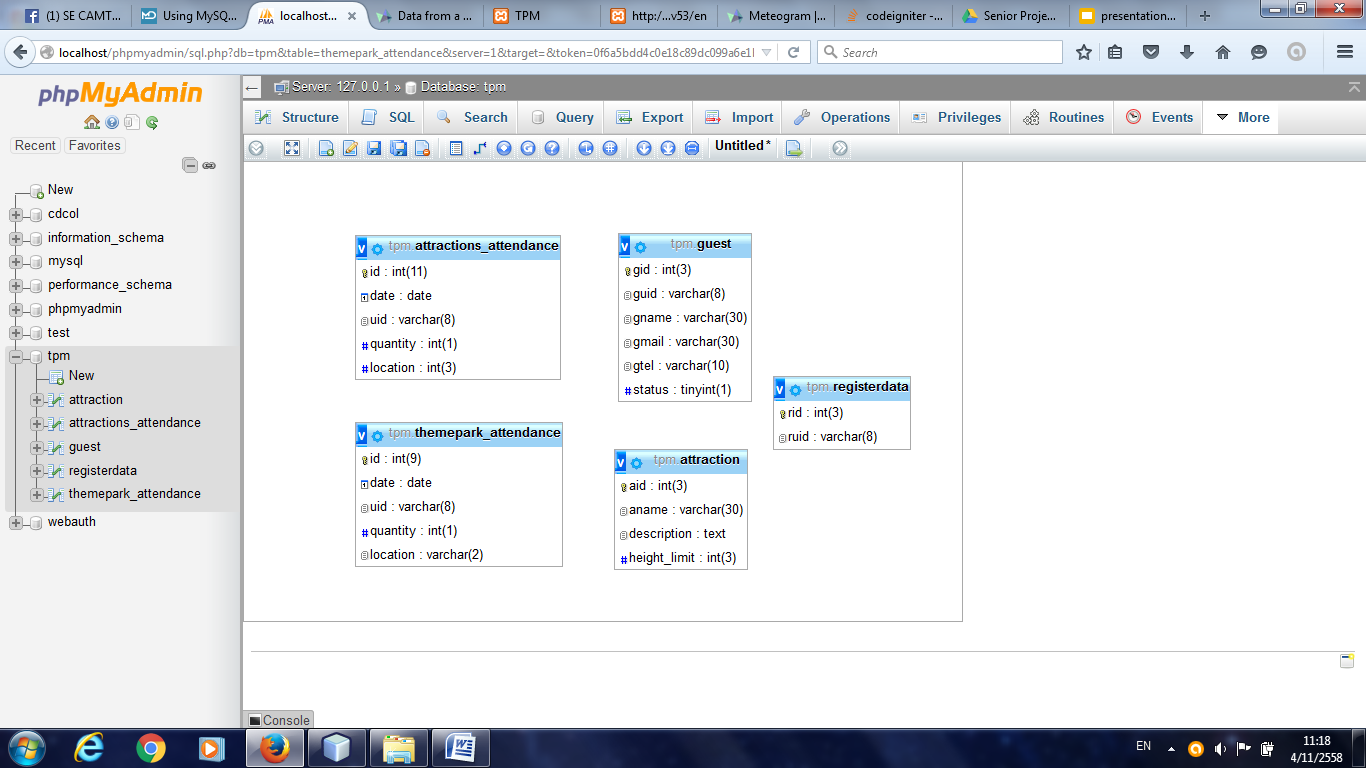


## SD-13: URS-13: Admin can view a report on numbers of guest attending each ride within one week range.



# Chapter Six | Database Design

## 6.1 ER Diagram



## 6.2 Table Description

### 6.2.1 attraction\_attendance

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name | Description | Type | Length | Primary Key | Foreign Key | Nullable |
| id | Id of attendance | integer | 11 | PK | 一 | Not Null |
| date | date of | date | - | 一 | 一 | Not Null |
| uid | uid of the guest | varchar | 8 | 一 | 一 | 一 |
| quantity | set default at 1 to get a number of guests i each date | int | 1 | 一 | 一 | Not Null |
| location | Location of the arduino device | varchar | 2 | 一 | 一 | Not Null |

### 6.2.2 themepark\_attendance

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name | Description | Type | Length | Primary Key | Foreign Key | Nullable |
| id | Id of attendance | integer | 11 | PK | 一 | Not Null |
| date | date of | date | - | 一 | 一 | Not Null |
| uid | uid of the guest | varchar | 8 | 一 | 一 | 一 |
| quantity | set default at 1 to get a number of guests i each date | int | 1 | 一 | 一 | Not Null |
| location | Location of the arduino device | varchar | 2 | 一 | 一 | Not Null |

### 6.2.3 guest

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name | Description | Type | Length | Primary Key | Foreign Key | Nullable |
| gid | Id of the guest | integer | 3 | PK | 一 | Not Null |
| guid | UID of the guest | date | 8 | 一 | 一 | Not Null |
| gname | name of the guest | varchar | 30 | 一 | 一 | Not Null |
| gmail | email of the guest | int | 30 | 一 | 一 | Not Null |
| gtel | Location of the arduino device | varchar | 10 | 一 | 一 | Not Null |
| status | indicate the status of the guest | tinyint | 1 | 一 | 一 | Not Null |

### 6.2.4 attraction

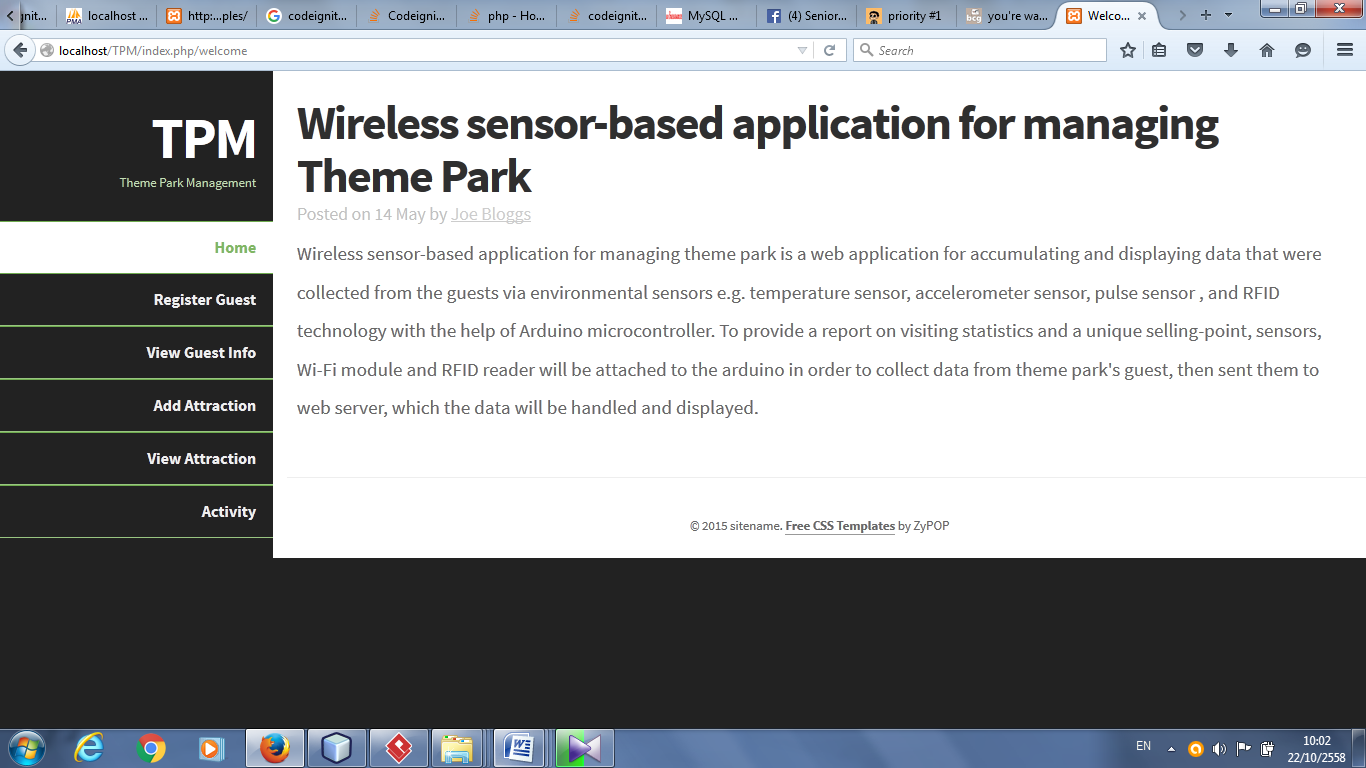
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name | Description | Type | Length | Primary Key | Foreign Key | Nullable |
| aid | Id of attendance | integer | 3 | PK | 一 | Not Null |
| aname | date of | date | 30 | 一 | 一 | Not Null |
| description | uid of the guest | text | 350 | 一 | 一 | 一 |
| height\_limit | set default at 1 to get a number of guests i each date | int | 3 | 一 | 一 | 一 |

### 6.2.5 registerdata

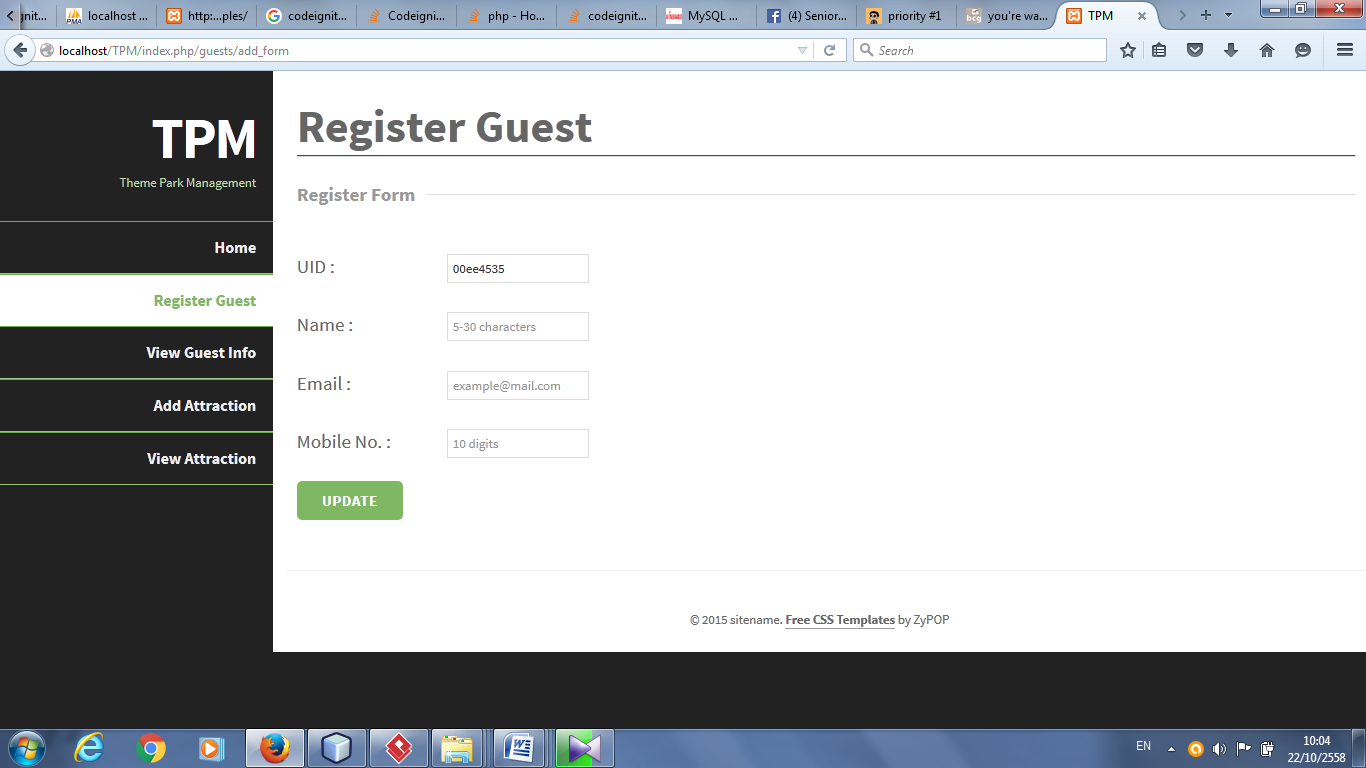
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name | Description | Type | Length | Primary Key | Foreign Key | Nullable |
| rid | Id of the UID that is read by the reader | integer | 3 | PK | 一 | Not Null |
| ruid | UID of the guest | date | 8 | 一 | 一 | Not Null |

# Chapter Seven | UI Design

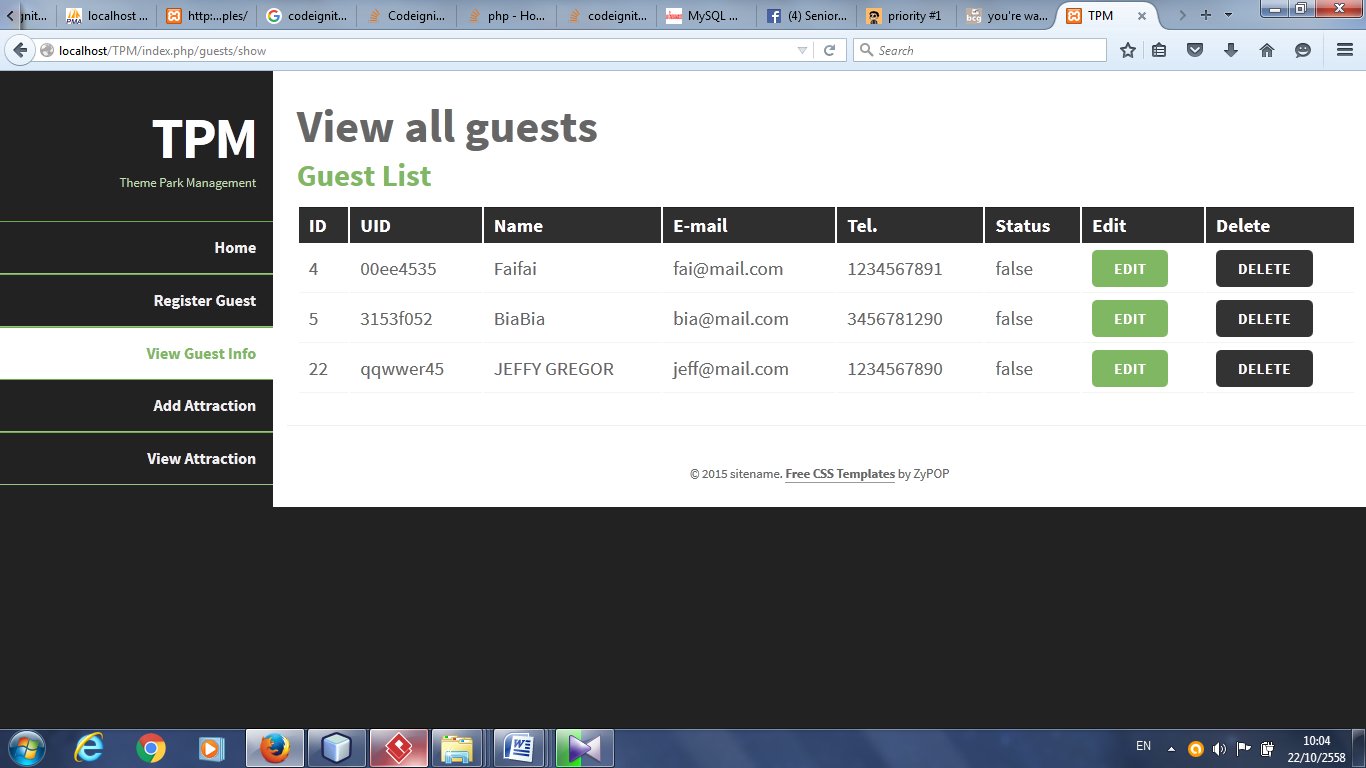
## UI-01: Home page



## UI-02: Guest register form



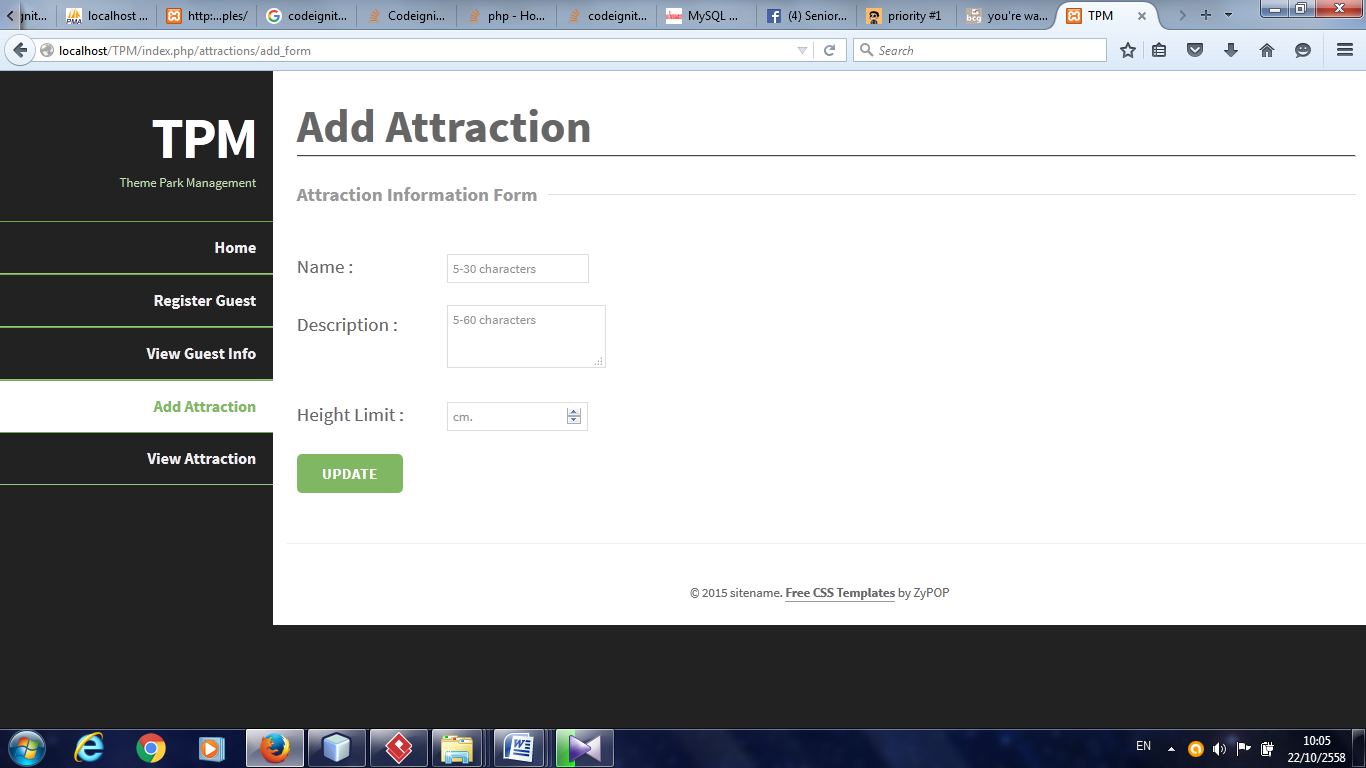
## UI-03: Guest list



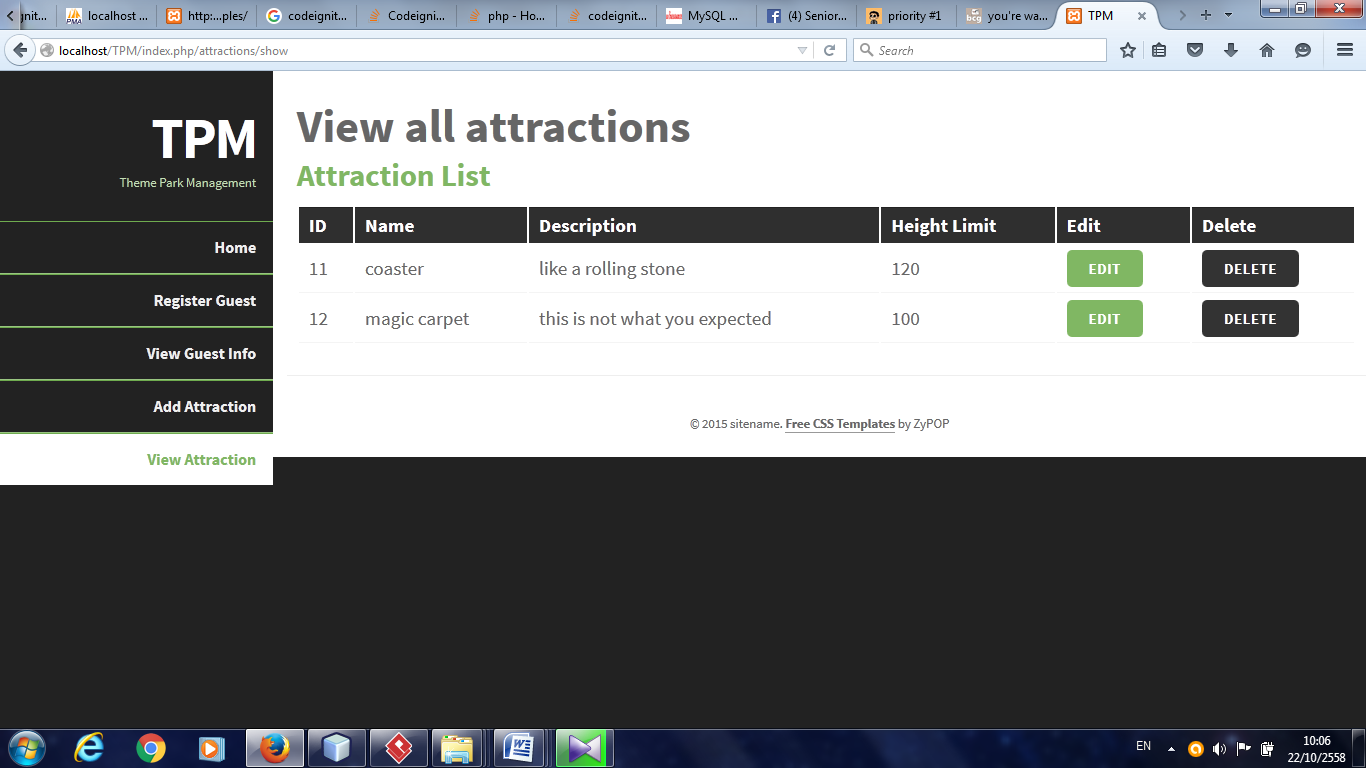
## UI-04: Guest edit form

## 

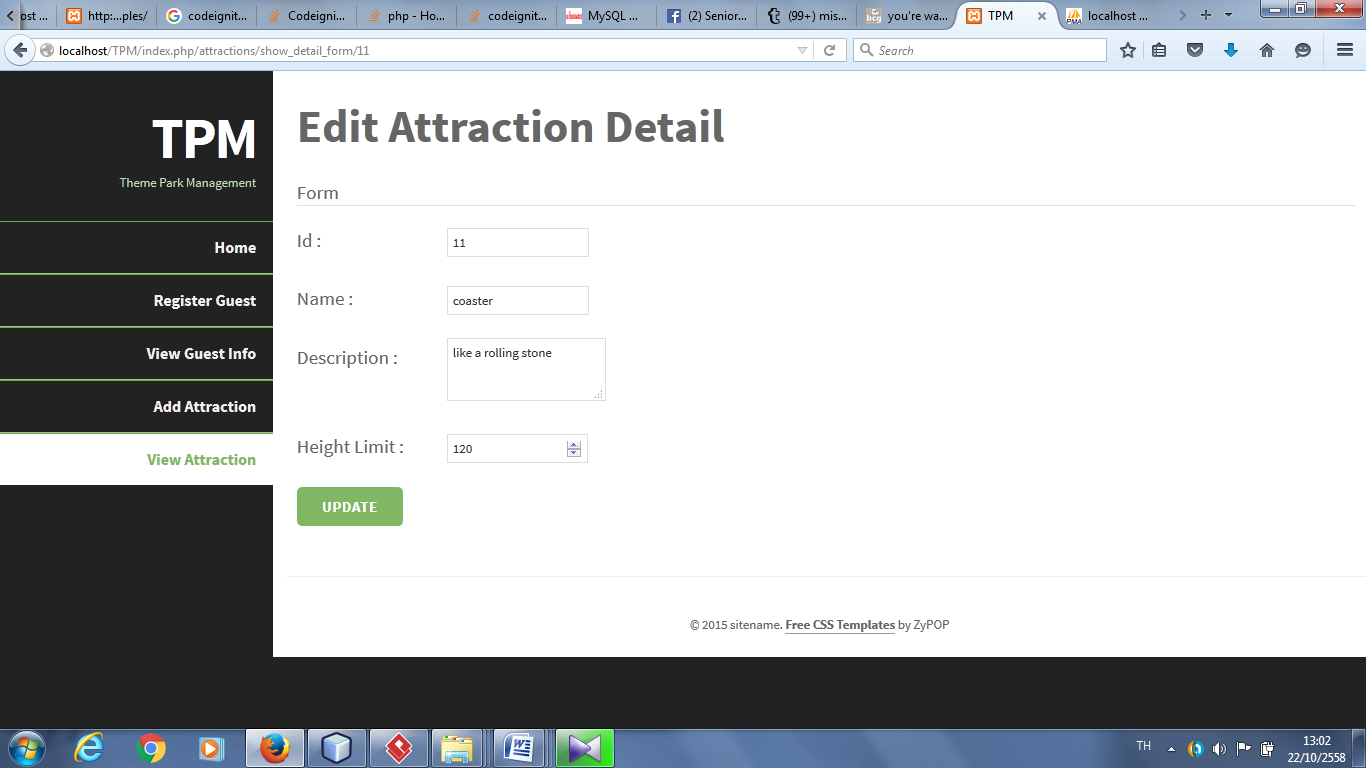
## UI-05: Attraction form



## UI-06: Attraction list



## UI-07: Attraction edit form



## UI-08: Activity line chart

## 

## 

# Chapter Eight | Wiring Schemes

The device consists of the following part:

* Arduino UNO
* Mifare RFID-MC522
* ESP8266 - 01
* Capacitor
* Regulator 5V -> 3.3V

## 8.1 Pinout

* **8.1.1. Mifare RFID-MC522**

|  |  |
| --- | --- |
| **MFRC522** | |
| **pinout** | **Arduino port** |
| SDA | D10 |
| SCK | D13 |
| MOSI | D11 |
| MISO | D12 |
| IRQ | none |
| GND | GND |
| RST | D9 |
| 3.3 V | 3.3V |

* **ESP8266 - 01**

|  |  |
| --- | --- |
| **ESP8266 - 01** | |
| **pinout** | **Arduino port** |
| VCC | 3.3V |
| RST | none |
| CH\_PD | 3.3V |
| TX | D6 |
| RX | D7 |
| GPIO | none |
| GPIO2 | none |
| GND | GND |