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](#_Toc428533171)

[Chapter One | Introduction 2](#_Toc428533172)

[1.1. Purpose 2](#_Toc428533173)

[1.2. Software Scope 2](#_Toc428533174)

[1.3. Acronyms and Definitions 2](#_Toc428533175)

[Chapter Two | System Architecture 4](#_Toc428533177)

[Chapter Three | Detailed Design 5](#_Toc428533178)

[3.1. Class Diagram Description 5](#_Toc428533179)

[Chapter Four | Class Description 7](#_Toc428533181)

[4.1. Class-01: Attraction 7](#_Toc428533182)

[4.2. Class-02: Guest 8](#_Toc428533183)

[4.3. Class-03: RegisterData 9](#_Toc428533184)

[4.4. Class-04: Attractions 9](#_Toc428533185)

[4.5. Class-05: Guests 10](#_Toc428533186)

[4.6. Class-06: Postdata 11](#_Toc428533187)

[4.7. Class-18: attraction\_form 12](#_Toc428533188)

[4.8. Class-19: attraction\_edit\_form 12](#_Toc428533189)

[4.9. Class-20: attraction\_view 12](#_Toc428533190)

[4.10. Class-21: guest\_form 12](#_Toc428533191)

[4.11. Class-22: guest\_edit\_form 13](#_Toc428533192)

[4.12. Class-23: guest\_view 13](#_Toc428533193)

[Chapter Five | Sequence Diagram 14](#_Toc428533194)

[5.1 Sequence Diagram response to URS-01: Ticket seller can register guest into the system. 14](#_Toc428533195)

[5.2 Sequence Diagram response to URS-02: Ticket seller can view guest information from the system 15](#_Toc428533196)

[5.3 Sequence Diagram response to URS-03: Ticket seller can edit guest information from the system. 16](#_Toc428533197)

[5.4 Sequence Diagram response to URS-04: Ticket seller can delete guest information from the system. 17](#_Toc428533198)

[5.5 Sequence Diagram response to URS-09: Admin can view attraction information from the system. 18](#_Toc428533199)

[5.6 Sequence Diagram response to URS-10: Admin can add attraction information into the system. 19](#_Toc428533200)

[5.7 Sequence Diagram response to URS-11: Admin can edit attraction information from the system. 20](#_Toc428533201)

[5.8 Sequence Diagram response to URS-12: Admin can delete attraction information from the system. 21](#_Toc428533202)

[Chapter Six | ER Diagram 22](#_Toc428533203)

# 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

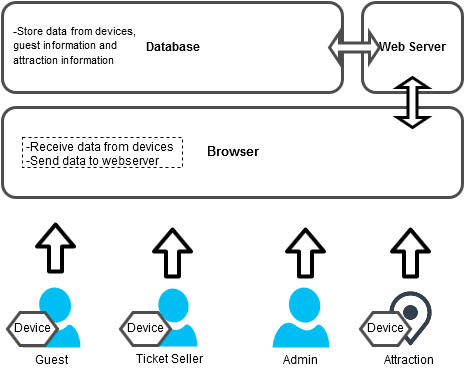


Figure 1: System architecture

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

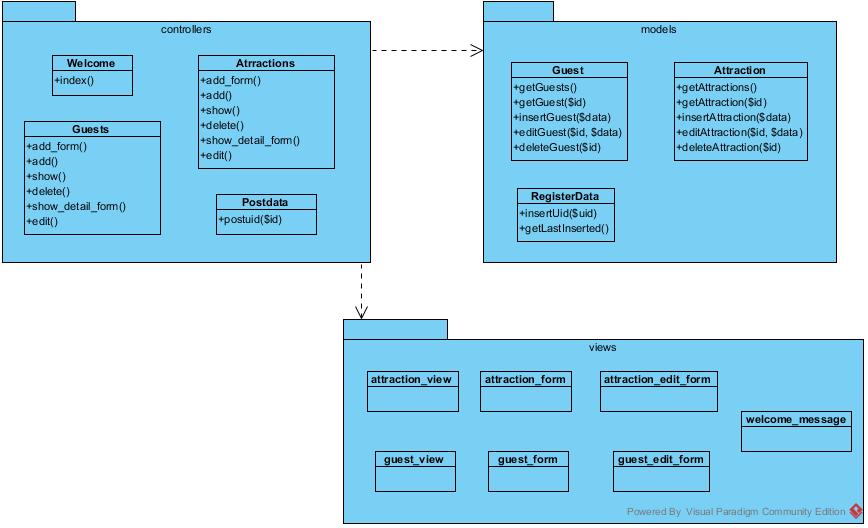
* **Arduino:** The arduino connects to RFID module and Wi-Fi module. It’ll be deployed alongside with stations, entrance gate, exit gate rides’ entrance and ticket seller boot. The main task is to send data from arduino to web server. The person who will use this are guest and ticket seller.
* **Web 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

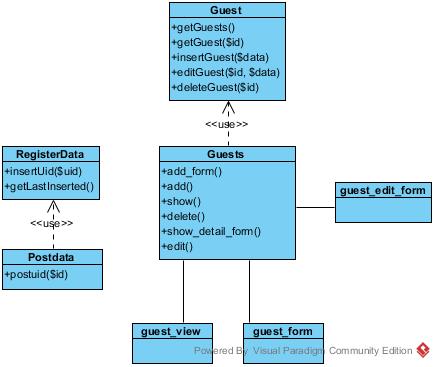
## Class Diagram Description

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.

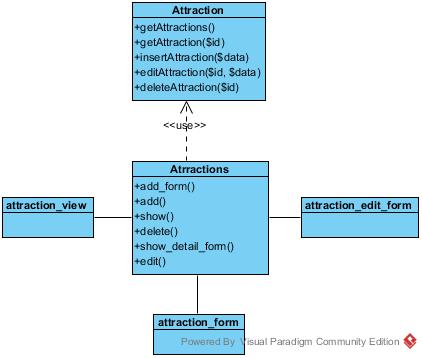
### Wireless sensor-based application System overall class diagram



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



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



# Chapter Four | Class Description

## Class-01: Attraction



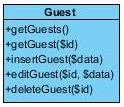
* **Description:**

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 : int** | **Object** |
| 3 | **insertAttraction(data)** | **This method is to insert an attraction into the database** | **data : array** | **Object** |
| 4. | **editAttraction(id, data)** | **This method is to update an attraction that has the same id as the parameter** | **id : int** | **Object** |
| 5 | **deleteAttraction(id)** | **This method is to get an attraction that has the same id as the parameter** | **id : int** | **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** |

## 

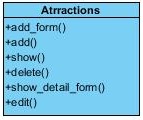
## Class-03: RegisterData

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** |

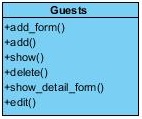
## Class-04: Attractions



* **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** |  |  |

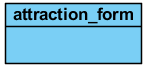
## Class-06: Postdata

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

* **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’** |  |  |

## Class-18: attraction\_form



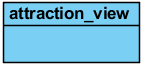
* **Description: This class is to provide input form and display error messages if any of the input is invalid.**
* **Attributes: -**
* **Methods: -**

## Class-19: attraction\_edit\_form



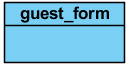
* **Description: This class is to provide an edit form and send the input to edit\_attraction for later operations.**
* **Attributes: -**
* **Methods: -**

## Class-20: attraction\_view



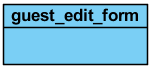
* **Description: This class is to display all attraction information in a table and provide edit and delete button for operations.**
* **Attributes: -**
* **Methods:**

## Class-21: guest\_form



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

## Class-22: guest\_edit\_form



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

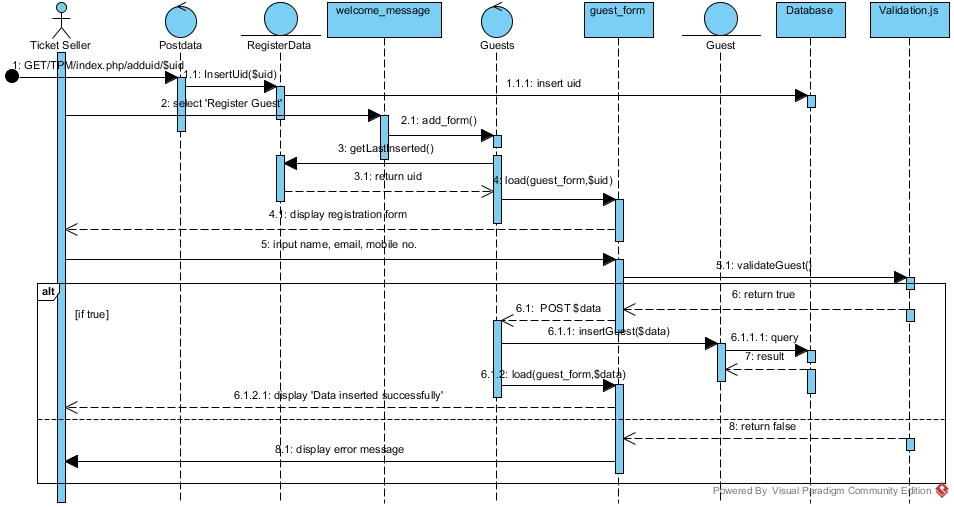
## Class-23: guest\_view



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

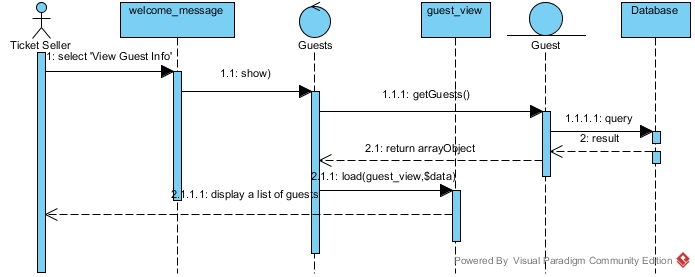
# Chapter Five | Sequence Diagram

## Sequence Diagram response to URS-01: Ticket seller can register guest into the system.

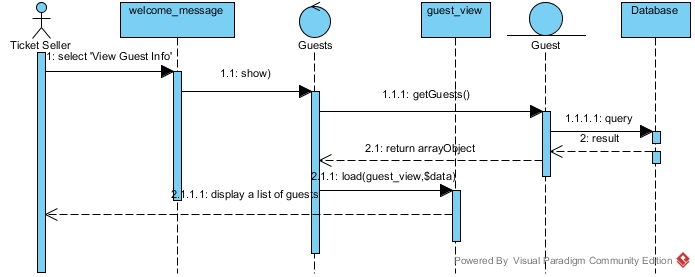


## 

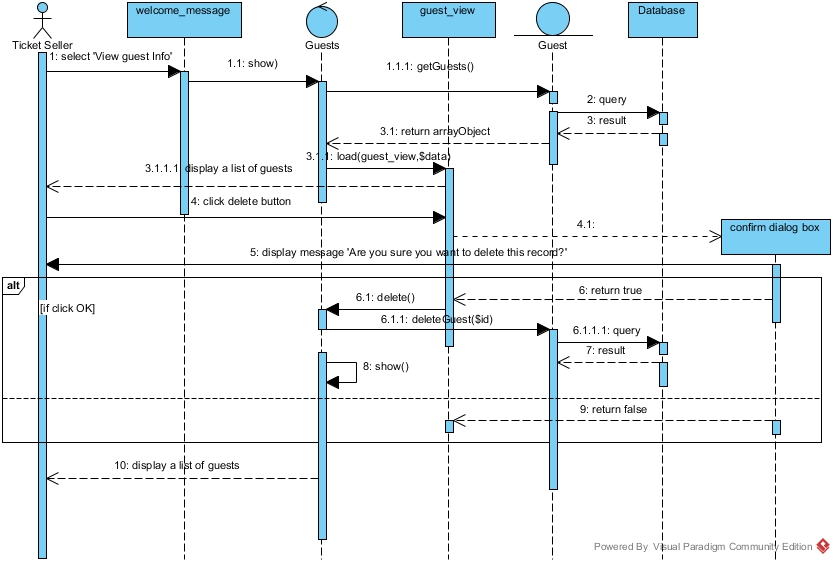
## Sequence Diagram response to URS-02: Ticket seller can view guest information from the system



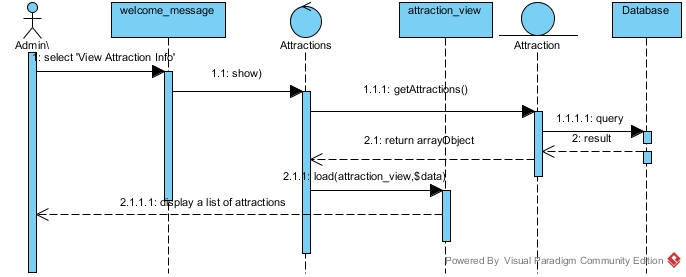
## Sequence Diagram response to URS-03: Ticket seller can edit guest information from the system.



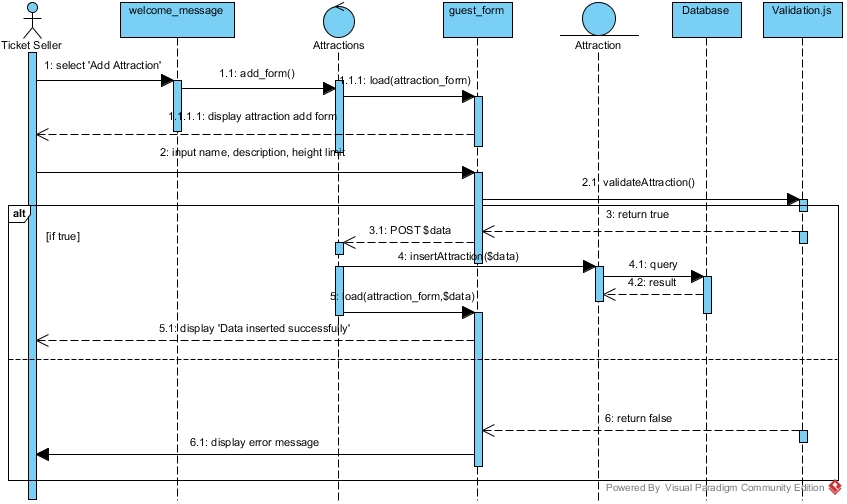
## Sequence Diagram response to URS-04: Ticket seller can delete guest information from the system.



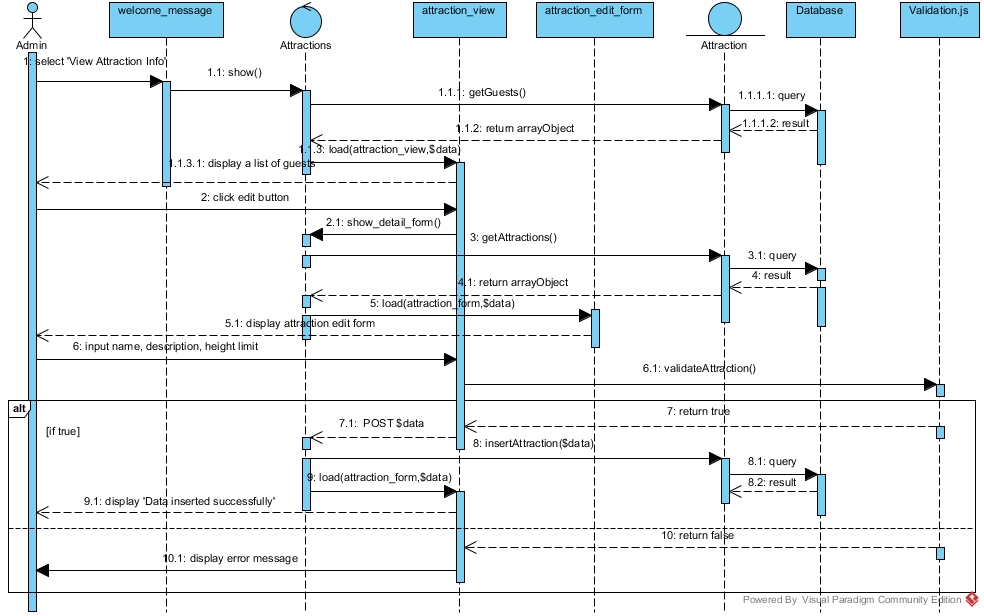
## Sequence Diagram response to URS-09: Admin can view attraction information from the system.



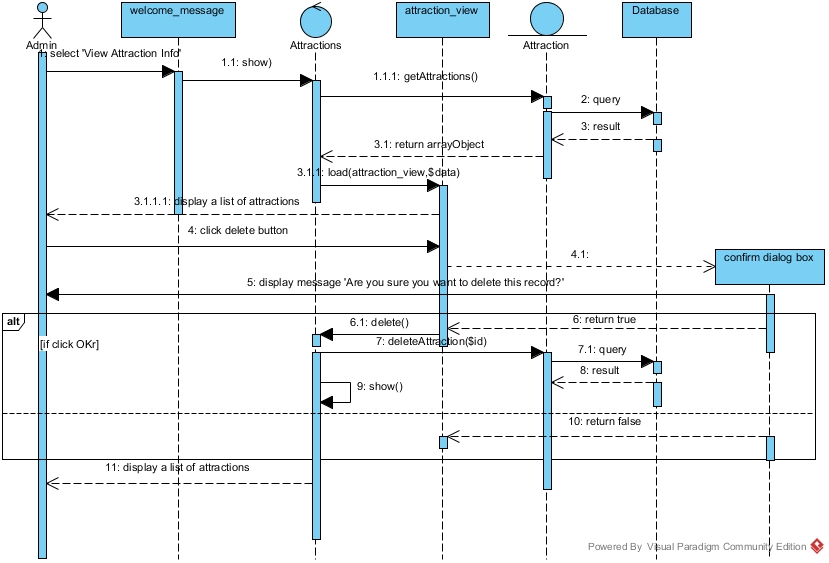
## Sequence Diagram response to URS-10: Admin can add attraction information into the system.



## Sequence Diagram response to URS-11: Admin can edit attraction information from the system.



## Sequence Diagram response to URS-12: Admin can delete attraction information from the system.



# Chapter Six | ER Diagram

