

**SECJ2203 SOFTWARE ENGINEERING**

**SYSTEM DOCUMENTATION**

**EVENT MANAGEMENT SYSTEM FOR K01,**

**KOLEJ TUN RAZAK**

**4th JULY 2024**

**FACULTY OF COMPUTING**

**LECTURER: DR. SIM HIEW MOI**

**SOFTWARE DESIGN DOCUMENTATION**

**PREPARED BY:**

|  |  |
| --- | --- |
| **GROUP MEMBERS** | **MATRIC NUMBER** |
| VENNISE NGOH YAN LENG | B23CS8032 |
| ARIEF NAJMI BIN BAKIYUDIN | A23CS8004 |
| OMAR ABDELMONEM HANAFY | A22EC4012 |
| AINA ZAFIRAH BINTI AHMAD MUZAMIR | A23CS8014 |
| SADIK AL MAHMUD | A20EC4049 |

**TABLE OF CONTENTS**

[**REVISION PAGE 4**](#_hvwi1g9cejx8)

[a. Overview 4](#_3nwe7t6mi46v)

[b. Target Audience 4](#_6ut0vujrfvzk)

[c. Project Team Members 4](#_bkg6ga93gg0f)

[d. Version Control History 5](#_ihyghnqa8uhs)

[**1.0 Introduction 6**](#_f3uhydwotvcj)

[1.1 Purpose 6](#_apn0gvfobl82)

[1.2 Scope 6](#_cs1m97efbba3)

[1.5 Overview 7](#_24t0s3pj7kue)

[**2.0 System Architectural Design 9**](#_p0e3smjmy6gp)

[2.1 Architecture Style and Rationale 9](#_gksgk3rh6um9)

[2.2 Architecture Model 9](#_f0afl72k7w6m)

[2.3 Use Case Diagram 11](#_2e3ztksfol5h)

[**3.0 Detailed Description of Components 12**](#_cgyh36wy8kqf)

[3.1 Complete Package Diagram 12](#_70pbl698844c)

[3.2 Component Model 12](#_u2ua9xch4zzg)

[3.3 Detailed Description 14](#_acyc86bvjqeh)

[3.3.1 Subsystem User Main Subsystem 14](#_qr5p7y7djstt)

[3.3.1.1 P001: Package User Requests 14](#_ciubq586yrf8)

[3.3.1.2 Class Diagram 14](#_5yg2exhbs60t)

[3.3.1.3 Sequence Diagrams 14](#_jztbxy7mbvuf)

[3.3.2 Subsystem Administrator Subsystem 15](#_71hh7t1xs7yg)

[3.3.2.1 P002: Package Event Management 15](#_a3vcljqa230l)

[3.3.2.2 Class Diagram 15](#_5h1ac4wlf1c8)

[Figure 3.6: Class diagram for Event Places Management 15](#_6hepmai4lmh2)

[Figure 3.7: Class diagram for Event Activities Management 15](#_nk84eoywl3y1)

[Figure 3.8: Class diagram for Event Promotion Package Management 15](#_zihka464igs)

[3.3.2.3 Sequence Diagrams 15](#_rewl47gqtwy)

[3.3.3 Subsystem Notification Subsystem 17](#_i6k5i6puryhd)

[3.3.3.1 P003:Package Notification Management 17](#_q0kq04toekgb)

[3.3.3.2 Class Diagram 17](#_t2gpeyxoqd1g)

[(System - generate()->Notification->send()->User) 17](#_8b4j2gn4dikc)

[Figure 3.18: Class diagram for Notification Management Package 17](#_68s9f44e3c2f)

[3.3.3.3 Sequence Diagrams 17](#_kys5cyu1vv1v)

[**4.0 Data Design 18**](#_iw1m79vtv0uv)

[4.1 Data Description 18](#_s9fjhgmjn10d)

[4.2 Data Dictionary 19](#_pxvj2imznkih)

[4.2.1 Entity: User 19](#_ypt5ua74844e)

[4.2.2 Entity: Admin User 20](#_la95rw3n4ysc)

[4.2.3 Entity: Admin User Permission 21](#_giasu12ycy2w)

[4.2.4 Entity: Page Tab 22](#_kwfnspe6dxoy)

[4.2.5 Entity: Sub Tab 23](#_l6w23vyi39j4)

[4.2.6 Entity: Event 24](#_59kg6qy21ik8)

[4.2.7 Entity: Place 25](#_2mut84udaw29)

[4.2.8 Entity: Activity 26](#_y0uuct6x1b22)

[4.2.9 Entity: Header 27](#_giasu12ycy2w)

[4.2.10 Entity: Gallery 28](#_9o636lavv3a5)

[4.2.11 Entity: Facilities 28](#_pkkuza5gxfcq)

[4.2.12 Entity: Opportunities 29](#_whzykz1q7nq5)

[4.2.13 Entity: Package 30](#_hvc9kr77k1pu)

[**5.0 User Interface Design 31**](#_blp63nig9ov8)

[5.1 Overview of User Interface 31](#_oun3m1zbgcw8)

[5.2 Screen Images 31](#_cpzwtte5oiki)

[**6.0 Requirements Matrix 32**](#_iwjfrioznddu)

[**7.0 Appendices 33**](#_emm9h5vdfcfe)

# 

# **REVISION PAGE**

## **Overview**

The current edition of the Software Design Document (SDD) for the Event Management System for Kolej Tun Razak offers an extensive overview of the system's design and structure. This version incorporates comprehensive illustrations, use cases, and data descriptions that encompass every aspect of the system's operation. The purpose of the document is to give the development team a clear and useful framework to operate within during the implementation phase, allowing for a thorough understanding and smooth integration of all the parts.

## **b.** **Target Audience**

The Software Design Documentation (SDD) is intended for project stakeholders, including developers, administrators, and users involved in the development, deployment, and use of the Event Management System for Kolej Tun Razak.

## **c.** **Project Team Members**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Task** | **Status** |
| Omar Abdelmonem | Team Member | 3.3 and 5.2 | Complete |
| Aina Zafirah | Team Member | 3.2 | Complete |
| Vennise Ngoh | **Team Leader** | 1,2,4,5.1,6 | Complete |
| Arief Najmi | Team Member | 3.2 | Complete |
| Sadik Al Mahmud | Team Member | 3.1 |  |

## 

## **d.** **Version Control History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Primary Author(s)** | **Description of Version** | **Date Completed** |
| 0.0.1 | Arief Najmi Bin Bakiyudin | Created Document Template | 29/6/2024 |
| 0.0.2 | Vennise Ngoh Yan Leng | Complete Introduction | 29/6/2024 |
| 0.0.3 | Vennise Ngoh Yan Leng | Complete System Architectural Design | 29/6/2024 |
| 0.0.4 | Vennise Ngoh Yan Leng | Complete Data Design | 29/6/2024 |
| 0.0.5 | Vennise Ngoh Yan Leng | Complete Layout for Detailed Description of Component | 2/7/2024 |
| 0.0.6 | Venise Ngoh Yan Leng | Complete Requirement Matrix | 2/7/2024 |
| 0.0.7 | Sadik | Complete 3.3 | 4/7/2024 |

# **1.0** **Introduction**

## **1.1** **Purpose**

This Software Design Document (SDD) describes the architecture and detailed design of the Event Management System. It is intended for software developers, project managers, and stakeholders who need to understand the system’s design while implementation.

## **1.2** **Scope**

The software product is to enhance the event requestment process for organizers and event fasciitis provider management by providing tools for scheduling, booking places, participant registration, event promotion and feedback collection. The Event Management System is designed to facilitate the planning, organization, and management of events such as conferences, workshops, and seminars. The system aims to provide a user-friendly interface for event organizers to manage event details, participants, and schedules efficiently.

**1.3** **Definitions, Acronyms and Abbreviation**

|  |  |
| --- | --- |
| **TERM** | **DEFINITION** |
| EMS | Event Management System |
| SDD | System Design Documentation - a document that describes the system architecture, components and interfaces in detail. |
| UI | User Interface - the point of human-computer interaction and communication in a device. This can include display screens, keyboards, a mouse and the appearance of a desktop. |
| UX | User Experience - the overall experience a user has while interacting with a product, including ease of use, efficiency and satisfaction. |
| DB | Database - an organized collection of structured information, or data, typically stored electronically in a computer system. |

**1.4** **References**

*\*\*\* Specify a complete list of references using a standardized reference format.*

## **1.5** **Overview**

This document serves as a comprehensive guide to the design and architecture of the Event Management System for Kolej Tun Razak. It is structured to provide clarity and detail on various aspects of the system, ensuring that developers, stakeholders, and other readers have a thorough understanding of its construction and functionality. The documentation is organized into the following sections:

1. System Architectural Design

This section outlines the overall architecture of the Event Management System, detailing the architectural style adopted, the key components, and the interactions between them. It provides a high-level view of how the system is structured to meet the specified requirements.

1. Detailed Description of Components

In this section, each component of the system is described in detail. This includes the purpose, functionality, and relationships of each component within the system. Diagrams and technical descriptions are used to elucidate the inner workings and the integration of these components.

1. Data Design

This part focuses on the design of the data structures and databases used within the system. It includes data models, schema definitions, and descriptions of how data is stored, accessed, and managed. This section ensures that the data handling mechanisms align with the system’s requirements and performance expectations.

1. User Interface Design

The user interface design section provides details on the layout, design principles, and user experience considerations for the system’s front end. Wireframes, mockups, and design guidelines are included to convey how users will interact with the system and navigate through its features.

1. Requirements Matric

This section maps the system requirements to specific components and features, ensuring traceability and completeness. It helps in verifying that all requirements are addressed in the design and provides a reference for testing and validation.

1. Appendices

Any additional information, such as supplementary diagrams, detailed explanations, or references, is included in the appendices. This section supports the main content and provides further clarification where needed.

# 

# **2.0** **System Architectural Design**

## **2.1** **Architecture Style and Rationale**

The architecture style chosen for the EMS is Layered Architecture Pattern, in which components/cde in this pattern are separated into several layers of subtask and they are arranged one above another. Each layer has its own main unique task to do and it is independent of one to another. Since each layer is independent, programmers can modify the code inside a layer without affecting others.

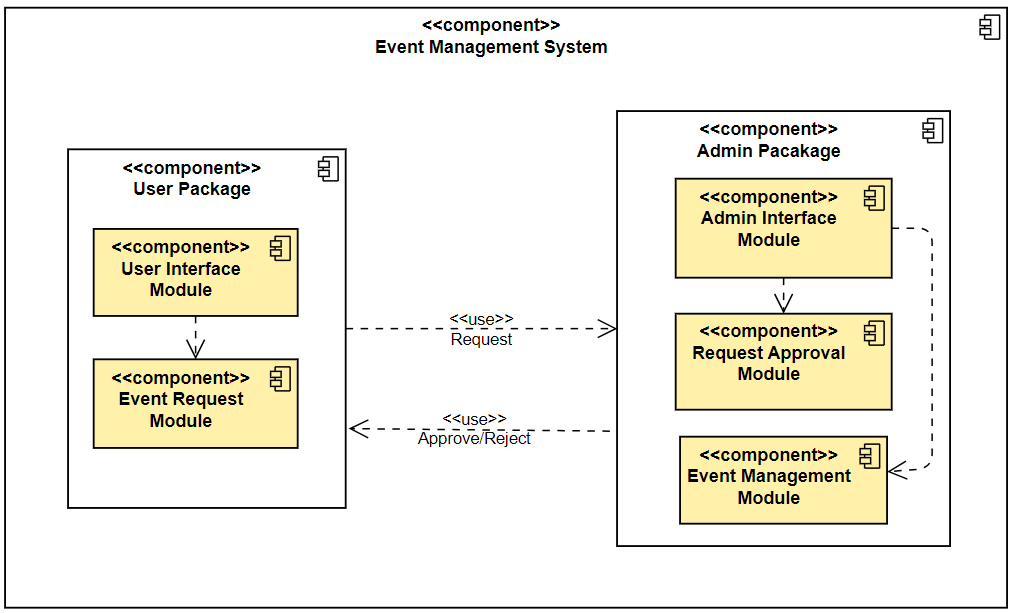
This style is chosen because it is scalable, flexible and easy for maintenance. All the layer can separate into 4 layer:

1. Presentation Layer : UI where the user sees.
2. Business Layer: Execute business logic from user interactions.
3. Application Layer: As a medium for passing data from Data Layer to Business Layer and vice versa.
4. Data Layer: This is a database layer.

## **2.2** **Architecture Model**

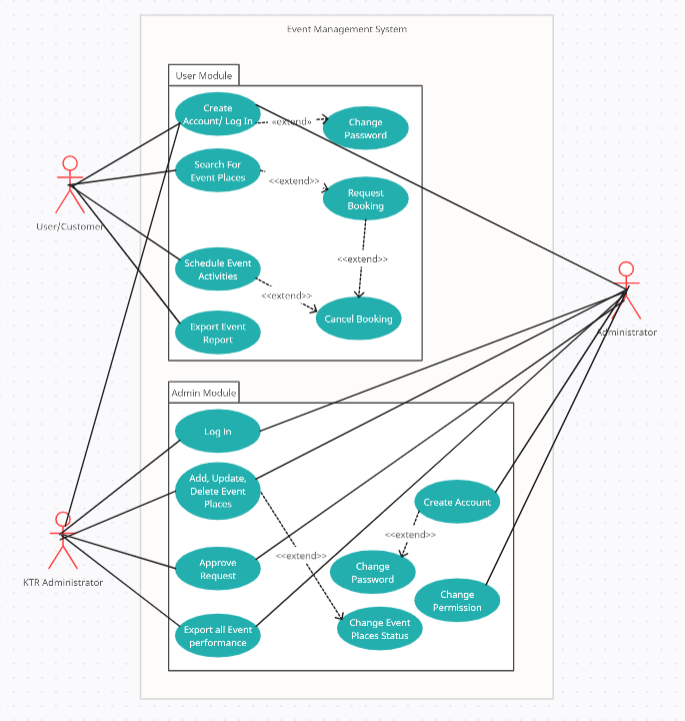
The system is divided into several level subsystems, each responsible for specific functionalities.

1. User Interface Subsystem: Handle UI and presentation logic.
2. Business Logic Subsystem: Contains the core functionalities and business rules of the system.
3. Data Access Subsystem: Manages data storage and retrieval operations.

**

***Figure 2.1: Component Model of Event Management System***

## **2.3** **Use Case Diagram**

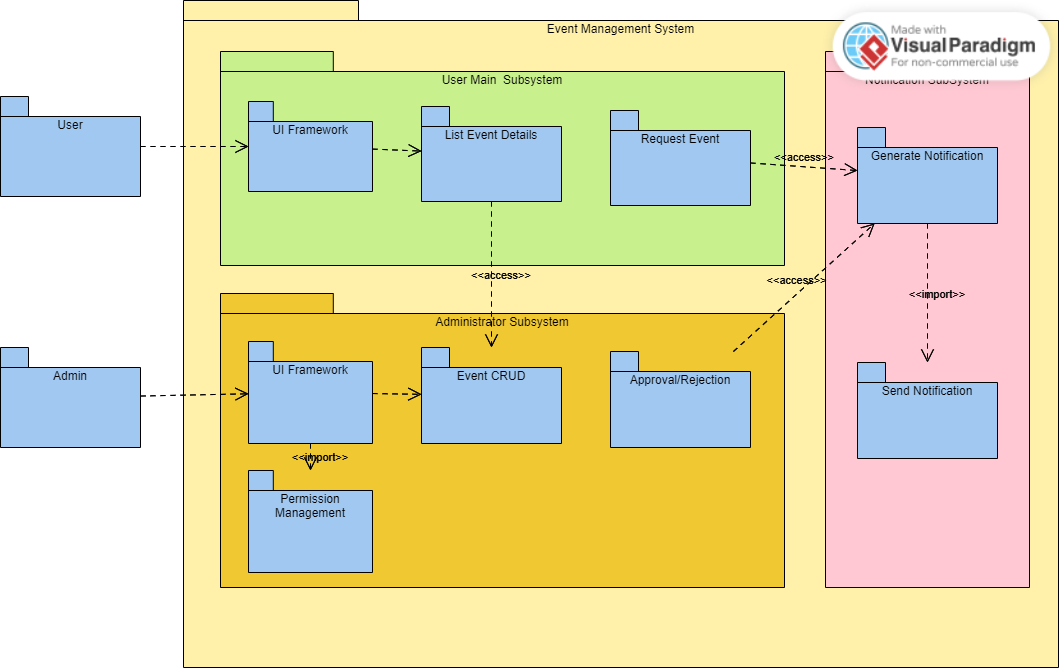


***Figure 2.2: Use Case Diagram of Event Management System***

# **3.0 Detailed Description of Components**

## **3.1** **Complete Package Diagram**

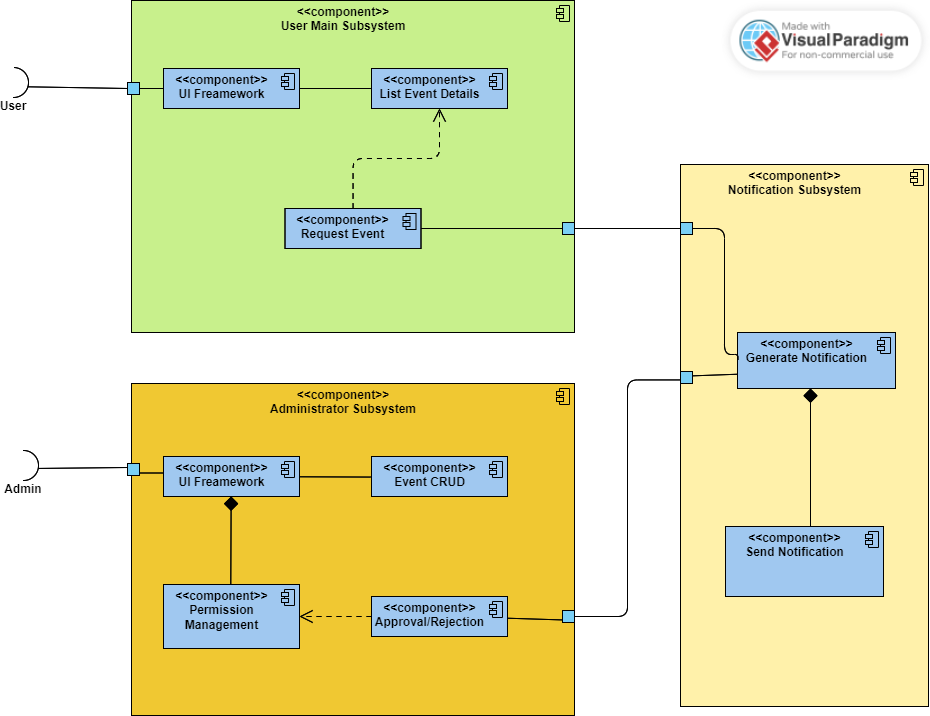
The Event Management System contains two main packages which are User Main Subsystem and Administrator Subsystem. The User Mian Subsystem is where users access requests for places for their event. While the Administrator Subsystem is the part where the Admin manages the offered places,activities and packages for users. The Notification Subsystem package is for generating notification and send notification to selected user/administrator.

**

***Figure 3.1: Subsystem of Event Management System***

## **3.2** **Component Model**

The component model of the Event Management System (EMS) provides a high-level overview of the major subsystems and how they interact to achieve the overall functionality of the system. This model is crucial for understanding the system's structure and the responsibilities of each subsystem. The component model ensures that the EMS is modular, scalable, and maintainable. Each subsystem can be developed, tested, and maintained independently while still integrating seamlessly with the rest of the system. This modular approach facilitates easier updates, enhancements, and troubleshooting.

**

***Figure 3.2: Component Diagram of Event Management System***

## 

## **3.3** **Detailed Description**

### **3.3.1** **Subsystem User Main Subsystem**

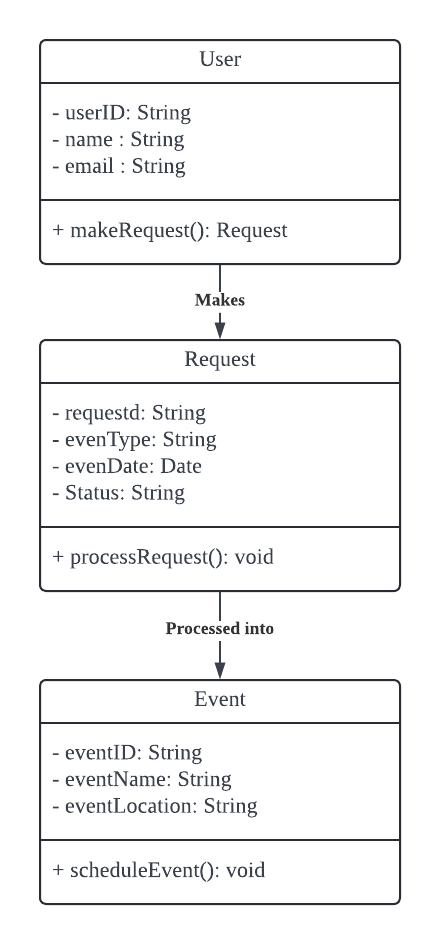
This is where the user lands on our system. Users will be able to request for booking event places, activities here.

#### **3.3.1.1 P001: Package User Requests**

This package handles user requests for booking event places, activities or promotion packages.

#### **3.3.1.2 Class Diagram**

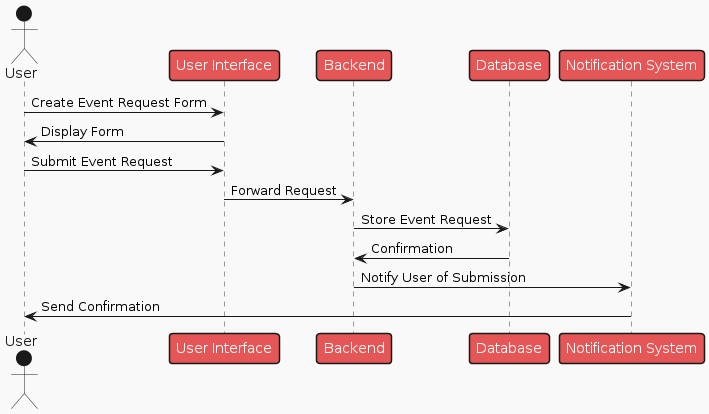
(User -> request()-> Event)



***Figure 3.3: Class diagram for User Request***

#### **3.3.1.3 Sequence Diagrams**

a) SD001: Sequence diagram for Create New Event Request



**Figure 3.4: Sequence Diagram of Create New Event Request**

b) SD002: Sequence diagram for Cancel Event Request



**Figure 3.5: Sequence Diagram of Cancel Event Request**

### **3.3.2 Subsystem Administrator Subsystem**

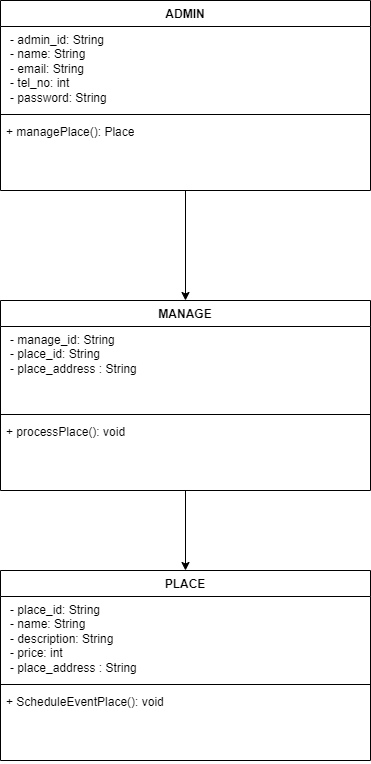
This is where the administrator lands, they can create, update, delete an event place, activities here. They also can reject/approve the user request for booking a place or activites.

#### **3.3.2.1 P002: Package Event Management**

This package manages the event places, activities, and promotion package.

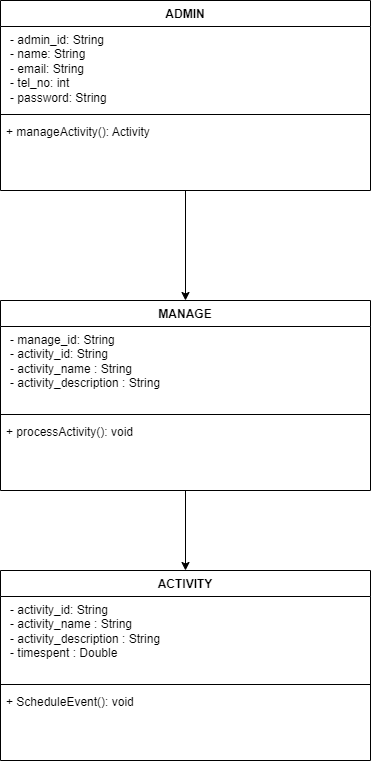
#### **3.3.2.2 Class Diagram**

(admin - manage()->Place)



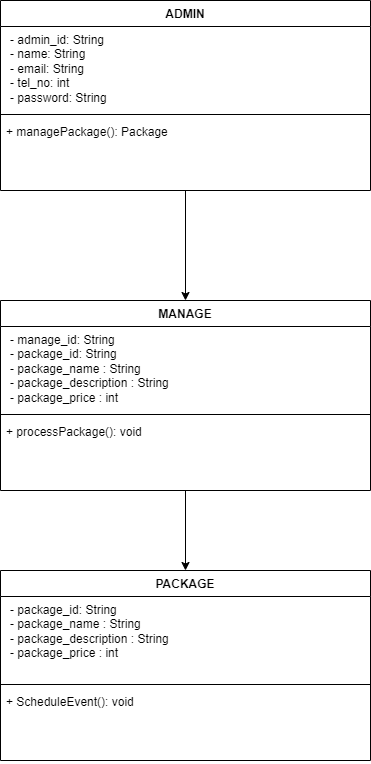
#### ***Figure 3.6: Class diagram for Event Places Management***

(admin - manage()->Activity)



#### ***Figure 3.7: Class diagram for Event Activities Management***

(admin - manage()->Package)

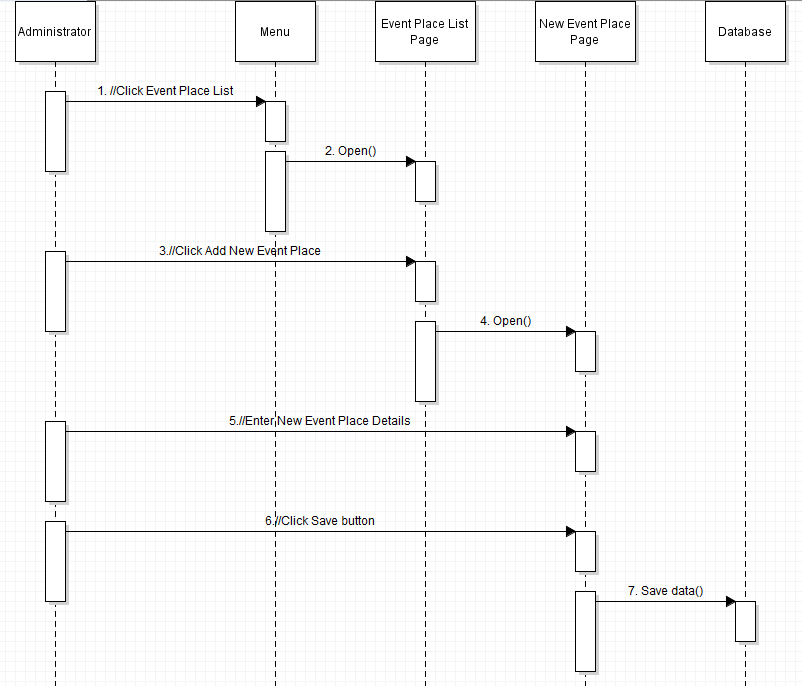


#### ***Figure 3.8: Class diagram for Event Promotion Package Management***

#### 

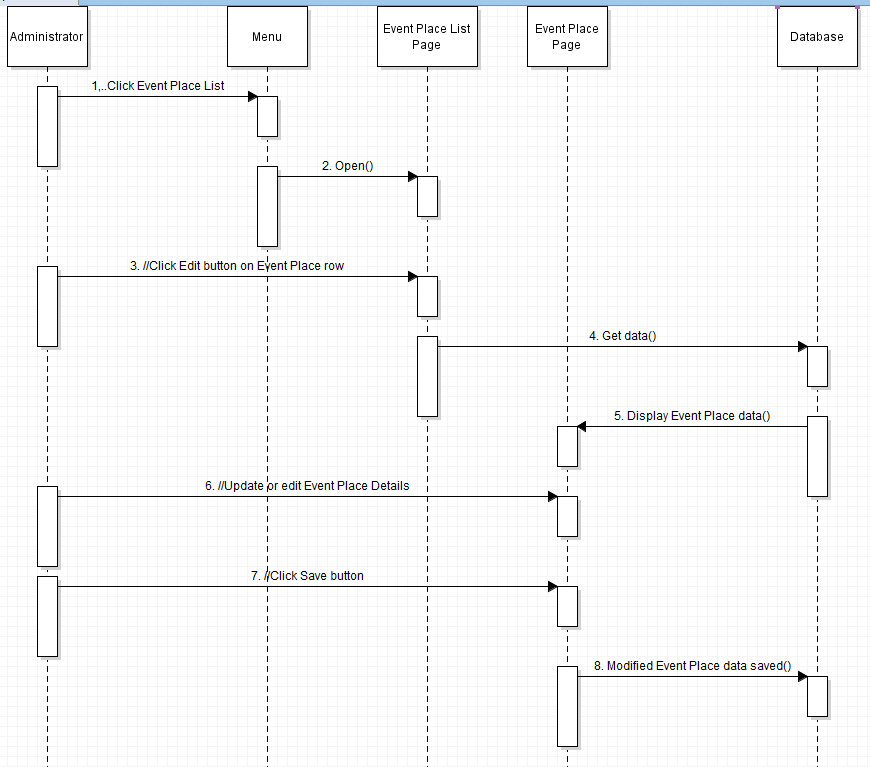
#### **3.3.2.3 Sequence Diagrams**

a) SD003: Sequence diagram for Create New Event Places



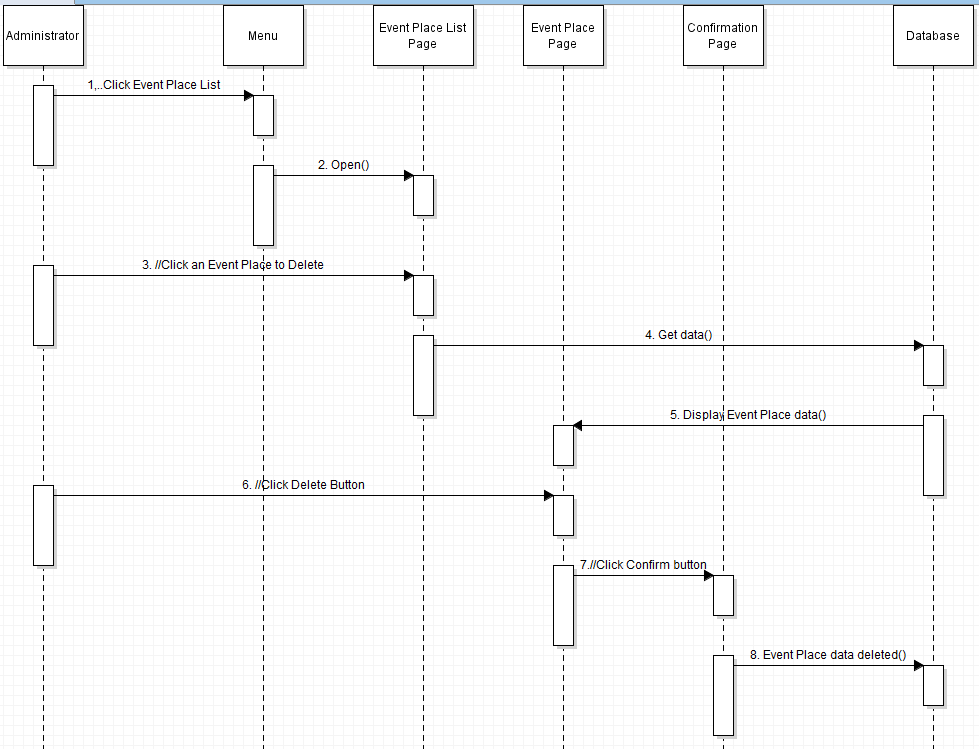
**Figure 3.9: Sequence Diagram of Create New Event Places**

b) SD004: Sequence diagram for Update Event Places



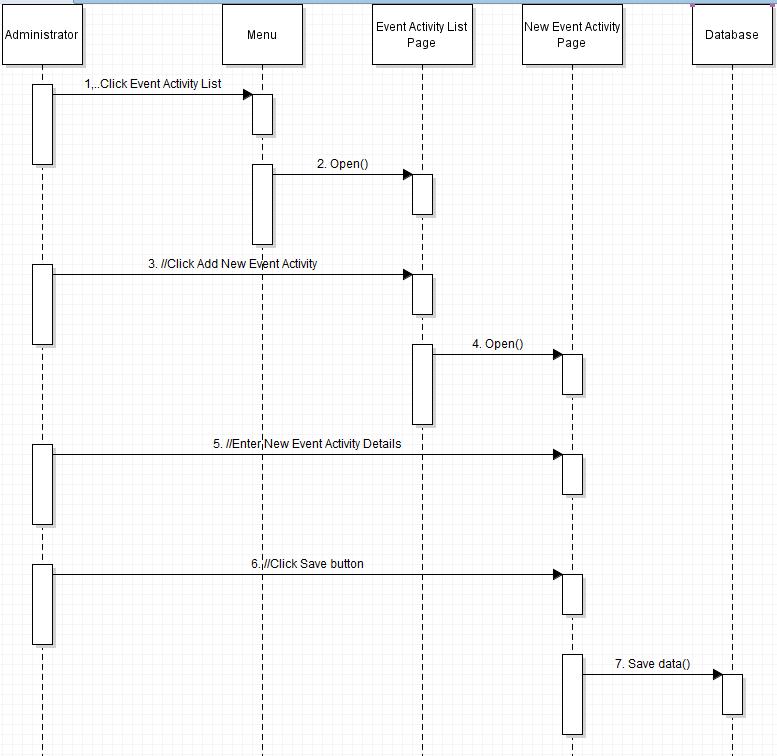
**Figure 3.10: Sequence Diagram of Update Event Places**

c) SD005: Sequence diagram for Delete Event Places



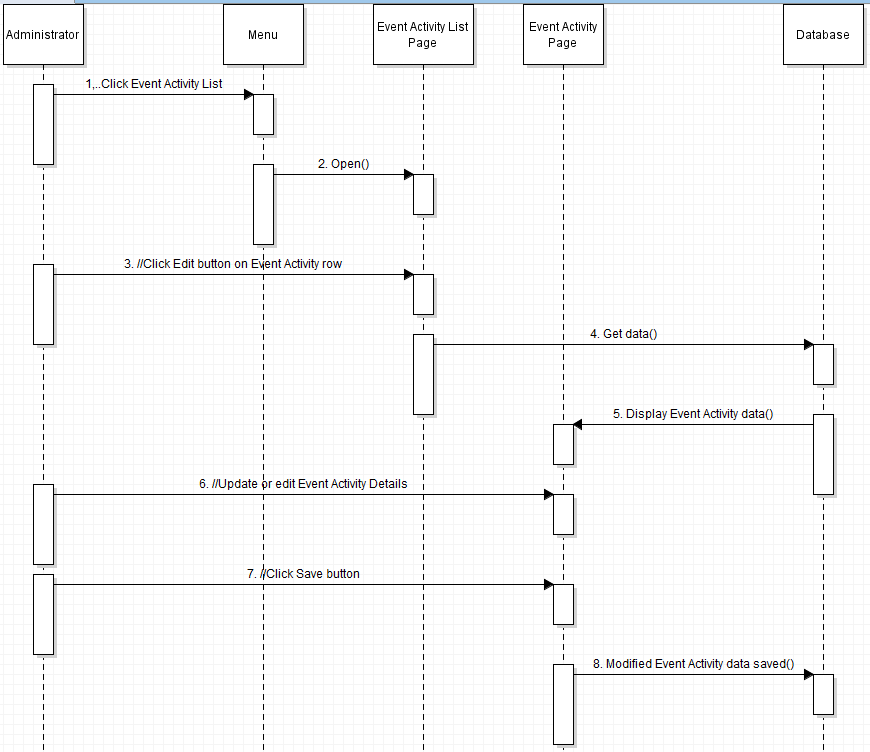
**Figure 3.11: Sequence Diagram of Delete Event Places**

d) SD006: Sequence diagram for Create New Event Activity



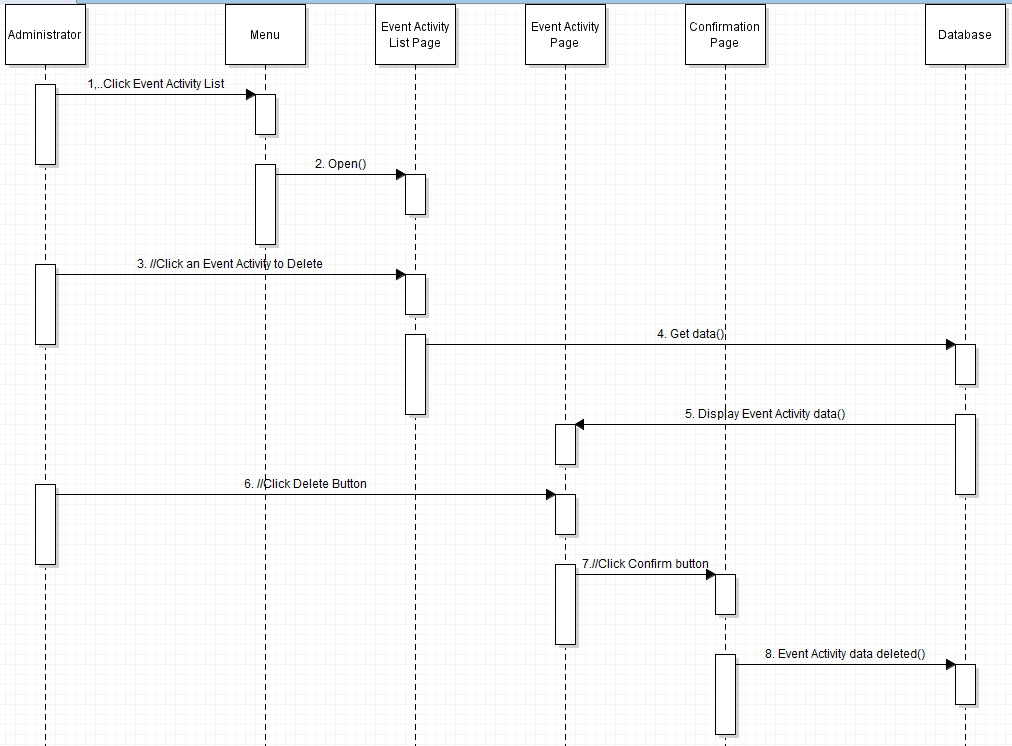
**Figure 3.12: Sequence Diagram of Create New Event Activity**

e) SD007: Sequence diagram for Update Event Activity



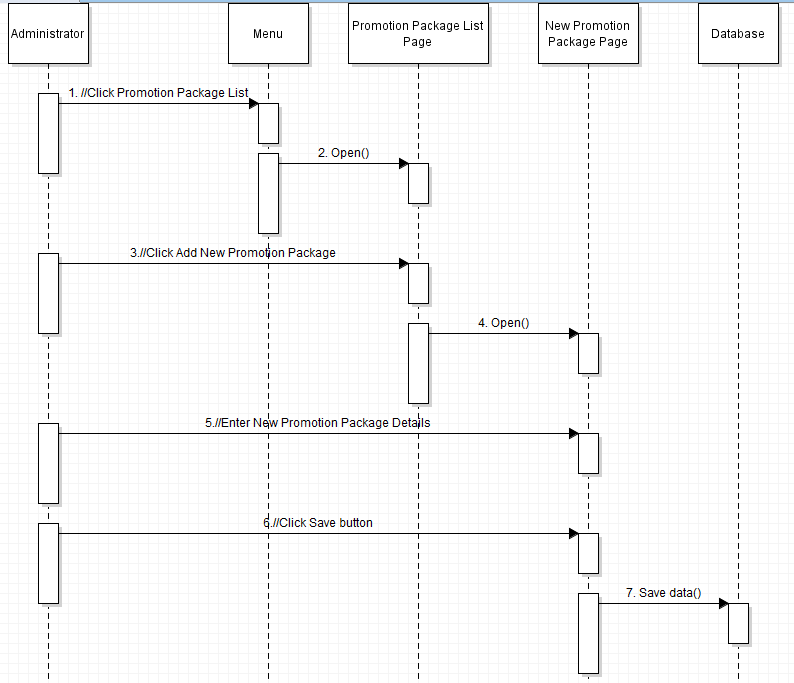
**Figure 3.13: Sequence Diagram of Update Event Activity**

f) SD008: Sequence diagram for Delete Event Activity



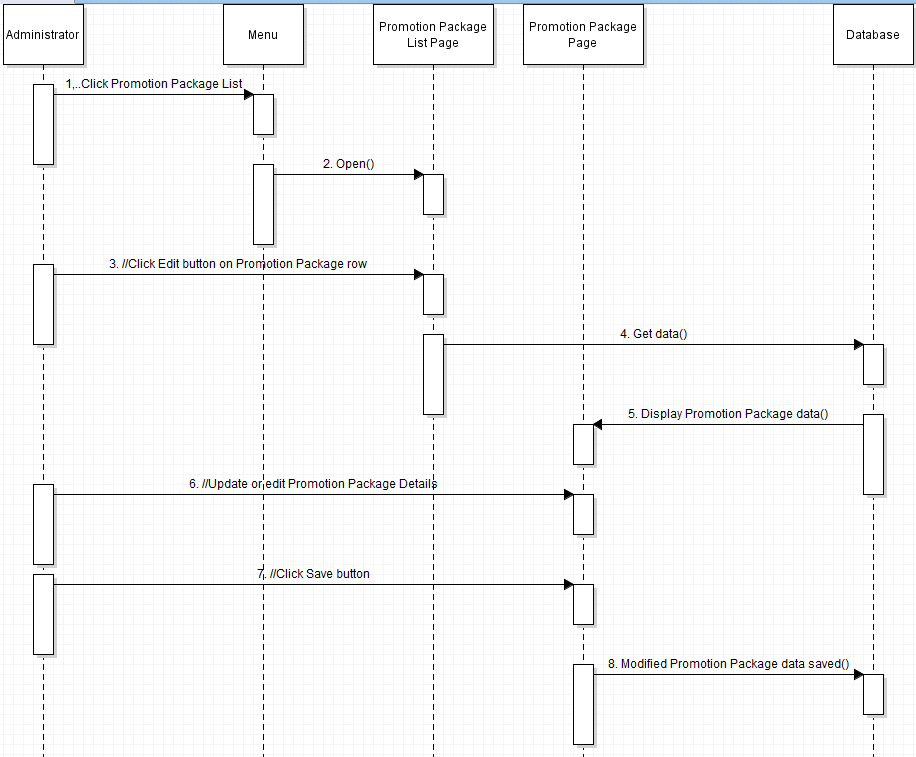
**Figure 3.14: Sequence Diagram of Delete Event Activity**

g) SD009: Sequence diagram for Create New Promotion Packages



**Figure 3.15: Sequence Diagram of Create New Promotion Package**

h) SD010: Sequence diagram for Update Promotion Package



**Figure 3.16: Sequence Diagram of Update Promotion Package**

i) SD011: Sequence diagram for Delete Promotion Package



**Figure 3.17: Sequence Diagram of Delete Promotion Package**

### **3.3.3 Subsystem Notification Subsystem**

This system generates and sends notifications to users and administrators about event requests, approval/rejection, event updates, reminders, and alerts.

#### **3.3.3.1 P003: Package Notification Management**

This package generates notifications and sends notifications to users and admins.

#### **3.3.3.3 Sequence Diagrams**

a) SD0012: Sequence diagram for Generate Notification

**Figure 3.19: Sequence Diagram of Generate Notification**

b) SD013: Sequence diagram for Send Notification

**Figure 3.20: Sequence Diagram of Send Notification**

# 

# **4.0 Data Design**

## **4.1** **Data Description**

All the data stored inside the Event Management System is stored into a relational table database named as per object and action. The entities in the database are described as below.

### 

|  |  |  |
| --- | --- | --- |
| **No** | **Entities** | **Description** |
| 1. | User | Stores main website user information |
| 2. | Admin User | Stores admin page user information |
| 3. | Admin User Permission | Stores each admin user permission in the admin page to access page tab or sub tab of a page tab. |
| 4. | Page Tab | Stores admin page page tab at the side menu. |
| 5. | Sub Tab | Stores admin page page tab sub tab in the side menu. |
| 6. | Event | Stores the data of the Event that is created in the admin page and shows in the main website. |
| 7. | Place | Stores the places declared for event places. |
| 8. | Activity | Stores the activity provided for the event. |
| 9. | Header | Direct each header id to each direction to get their data. |
| 10. | Gallery | Using header id to hash out or store in pictures to a place/ activity. |
| 11. | Facilities | Using header id to hash out or store facilities to a place/ activity. |
| 12. | Opportunities | Using header id to hash out or store opportunities description to a place/ activity. |
| 13. | Package | A table stores the package that combines Activity and Place with a lower price for a specific user. |

## **4.2** **Data Dictionary**

### **4.2.1 Entity: User**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the object User. |
| 2. | name | String | Stores a string of characters and numbers that identify the user name. |
| 3. | email | String | Stores a string of gmail based format of user for authentication use. |
| 4. | tel\_no | Number | Stores a series of numbers that can be used to contact users. |
| 5. | status | Char | Stores a single character that defines user account status. |
| 6. | password | String | encrypted strings for authentication use. |
| 7. | create\_date | DateTime | Stores a datetime data that defines the date and times the account created. |
| 8. | create\_by | String | Stores a string of username who created the account. |
| 9. | modify\_date | DateTime | Stores a datetime data that defines the last date and times the user modified their personal data. |
| 10. | modify\_by | String | Stores a string of username who last modified the user profile. |

### 

### **4.2.2 Entity: Admin User**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the object User. |
| 2. | name | String | Stores a string of characters and numbers that identify the user name. |
| 3. | email | String | Stores a string of gmail based format of user for authentication use. |
| 4. | tel\_no | Number | Stores a series of numbers that can be used to contact users. |
| 5. | status | Char | Stores a single character that defines user account status. |
| 6. | password | String | encrypted strings for authentication use. |
| 7. | create\_date | DateTime | Stores a datetime data that defines the date and times the account created. |
| 8. | create\_by | String | Stores a string of username who created the account. |
| 9. | modify\_date | DateTime | Stores a datetime data that defines the last date and times the user modified their personal data. |
| 10. | modify\_by | String | Stores a string of username who last modified the user profile. |

### 

### 

### **4.2.3 Entity: Admin User Permission**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the admin user permission |
| 2. | user\_id | String | Store the user’s id that related for this permission |
| 3. | page\_id | String | Store the page’s id for this permission |
| 4. | create | Number | If this data = 1, then the user can create data in this page\_id’s page. Else if it is 0, the user cannot create any data in this page. |
| 5. | update | Number | If this data = 1, then the user can modify data in this page\_id’s page. Else if it is 0, the user cannot modify any data in this page. |
| 6. | delete | Number | If this data = 1, then the user can delete data in this page\_id’s page. Else if it is 0, the user cannot delete any data in this page. |
| 7. | create\_date | DateTime | Stores a datetime data that defines the date and times the account created. |
| 8. | create\_by | String | Stores a string of username who created the account. |
| 9. | modify\_date | DateTime | Stores a datetime data that defines the last date and times the user modified their personal data. |
| 10. | modify\_by | String | Stores a string of username who last modified the user profile. |

### 

### 

### **4.2.4 Entity: Page Tab**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the page tab |
| 2. | name | String | Stores the page tab name that will shows at the side menu |
| 3. | file\_path | String | Stores the code file path inside the file. |
| 4. | sequence\_no | Number | Stores the number of sequences that will affect the arrangement on the side menu. |
| 5. | create\_date | DateTime | Stores a datetime data that defines the date and times the page tab. |
| 6. | create\_by | String | Stores a string of username who created the page tab. |
| 7. | modify\_date | DateTime | Stores a datetime data that defines the last date and times the user modified the page tab details. |
| 8. | modify\_by | String | Stores a string of username who last modified the page tab. |

### 

### 

### **4.2.5 Entity: Sub Tab**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the sub tab |
| 2. | header\_id | String | Stores the parent page tab id. |
| 3. | name | String | Stores the sub tab name that will shows under their parent page tab |
| 4. | file\_path | String | Stores the code file path inside the file. |
| 5. | sequence\_no | Number | Stores the number of sequences that will affect the arrangement on the side menu. |
| 6. | create\_date | DateTime | Stores a datetime data that defines the date and times the sub tab created. |
| 7. | create\_by | String | Stores a string of username who created the sub tab. |
| 8. | modify\_date | DateTime | Stores a datetime data that defines the last date and times the user modified the sub tab. |
| 9. | modify\_by | String | Stores a string of username who last modified the sub tab. |

### 

### 

### **4.2.6 Entity: Event**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the event. |
| 2. | name | String | Stores the event name |
| 3. | activity\_id | String | Stores the activity id if the event have any |
| 4. | place\_id | String | Stores the event place id |
| 5. | start\_date | date | Stores the event start date |
| 6. | end\_date | date | Stores the event end date |
| 7. | price | double | Stores the total prices (from places and activity or a package) need to be collected from the organizer. |
| 8. | color | String | Store the HEX code of color of the Event to make the data statistic beautiful. |
| 9. | status | Char | Store the event status. (A:Active,D:Delayed,C:Canceled) |
| 10. | create\_date | DateTime | Stores a datetime data that defines the date and times the event created. |
| 11. | create\_by | String | Stores a string of username who created the event. |
| 12. | modify\_date | DateTime | Stores a datetime data that defines the last date and times the user modified the event. |
| 13. | modify\_by | String | Stores a string of username who last modified the event. |

### 

### 

### **4.2.7 Entity: Place**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the place. |
| 2. | name | String | Stores the place name |
| 3. | size\_widthFeet | Number | Stores the width size in feet of the place. |
| 4. | size\_widthInches | Number | Stores the width size of the Inches of the place |
| 5. | size\_heightFeet | Number | Stores the height size in feet of the place. |
| 6. | size\_heightInches | Number | Stores the height size of the Inches of the place |
| 7. | address | String | Stores the address of the event place |
| 8. | price | double | Stores the price for the place |
| 9. | facilities\_header\_id | String | Stores the header id which will lead to the facilities database to get the facilities data. |
| 10. | galalery\_header\_id | String | Stores the header id which will lead to the gallery database to get the galleries data. |
| 11. | status | Char | Stores the status of the place. If it is not active, it will never show in the main page.  (A: Active, C: Canceled) |
| 12. | create\_date | DateTime | Stores a datetime data that defines the date and times the place created. |
| 13. | create\_by | String | Stores a string of username who created the place. |
| 14. | modify\_date | DateTime | Stores a datetime data that defines the last date and times the user modified the place. |
| 15. | modify\_by | String | Stores a string of username who last modified the place. |

### 

### **4.2.8 Entity: Activity**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the activity. |
| 2. | name | String | Stores the activity name |
| 3. | description | String | Describe the activity. |
| 4. | time\_spent | Double | Stores the time spent on the activity. |
| 5. | color | String | Store the HEX code of the color that represents the activity. |
| 6. | oppertunities\_header\_id | String | Stores the header\_id which will lead to the opportunities database to get the opportunities data. |
| 7. | price | Double | Stores the price for the activity. |
| 8. | gallery\_header\_id | String | Stores the header\_id which will lead to the gallery database to get the gallery data. |
| 9. | create\_date | DateTime | Stores a datetime data that defines the date and times the activity created. |
| 10. | create\_by | String | Stores a string of username who created the place. |
| 11. | modify\_date | DateTime | Stores a datetime data that defines the last date and times the user modified the activity. |
| 12. | modify\_by | String | Stores a string of username who last modified the activity. |

### 

### **4.2.9 Entity: Header**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the header. |
| 2. | total\_num | Number | Stores the total\_number of data can be get from this id |
| 3. | create\_date | DateTime | Stores a datetime data that defines the date and times the header created. |
| 4. | create\_by | String | Stores a string of username who created the header. |
| 5. | modify\_date | DateTime | Stores a datetime data that defines the last date and times the user modified the header. |
| 6. | modify\_by | String | Stores a string of username who last modified the header. |

### 

### 

### 

### 

### 

### 

### 

### **4.2.10 Entity: Gallery**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the image. |
| 2. | header\_id | String | Stores the header\_id for this image. |
| 3. | file\_name | String | Stores the file name of this image. |
| 4. | file\_path | String | Stores the actual file path of this image in the server. |
| 5. | create\_date | DateTime | Stores a datetime data that defines the date and times the activity created. |
| 6. | create\_by | String | Stores a string of username who created the place. |

### 

### 

### 

### **4.2.11 Entity: Facilities**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the facility. |
| 2. | header\_id | String | Stores the header\_id for this facility. |
| 3. | name | String | Stores the name of this facility. |
| 4. | icon | String | Stores fa-icon class code that will display for this facility. |
| 5. | create\_date | DateTime | Stores a datetime data that defines the date and times the facility was created. |
| 6. | create\_by | String | Stores a string of username who created the facility. |

### 

### 

### **4.2.12 Entity: Opportunities**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the opportunity. |
| 2. | header\_id | String | Stores the header\_id for this opportunity. |
| 3. | name | String | Stores the name of this opportunity. |
| 4. | icon | String | Stores fa-icon class code that will display for this opportunity. |
| 5. | sequence\_no | Number | Shows the sequence number that will affect the arrangement in the opportunity list. |
| 6. | create\_date | DateTime | Stores a datetime data that defines the date and times the opportunity created. |
| 7. | create\_by | String | Stores a string of usernames who created the opportunity . |

### 

### 

### 

### **4.2.13 Entity: Package**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Attribute** | **Data Type** | **Description** |
| 1. | unique\_id | String | uniquely identifies the package. |
| 2. | name | String | Stores the package name |
| 3. | activity\_id | String | Stores the activity related inside this package. |
| 4. | place | String | Stores the places that will be used inside this package. |
| 5. | price | Double | Stores the price for the package. |
| 6. | oppertunities\_header\_id | String | Stores the header\_id which will lead to the opportunities database to get the gallery data. |
| 7. | gallery\_header\_id | String | Stores the header\_id which will lead to the gallery database to get the gallery data. |
| 8. | description | String | Stores the description of the package. |
| 9. | time\_range | Double | Stores the time length for this package. |
| 10. | create\_date | DateTime | Stores a datetime data that defines the date and times the package created. |
| 11. | create\_by | String | Stores a string of username who created the package. |
| 12. | modify\_date | DateTime | Stores a datetime data that defines the last date and times the user modified the package. |
| 13. | modify\_by | String | Stores a string of username who last modified the package. |

### 

# **5.0 User Interface Design**

## **5.1 Overview of User Interface**

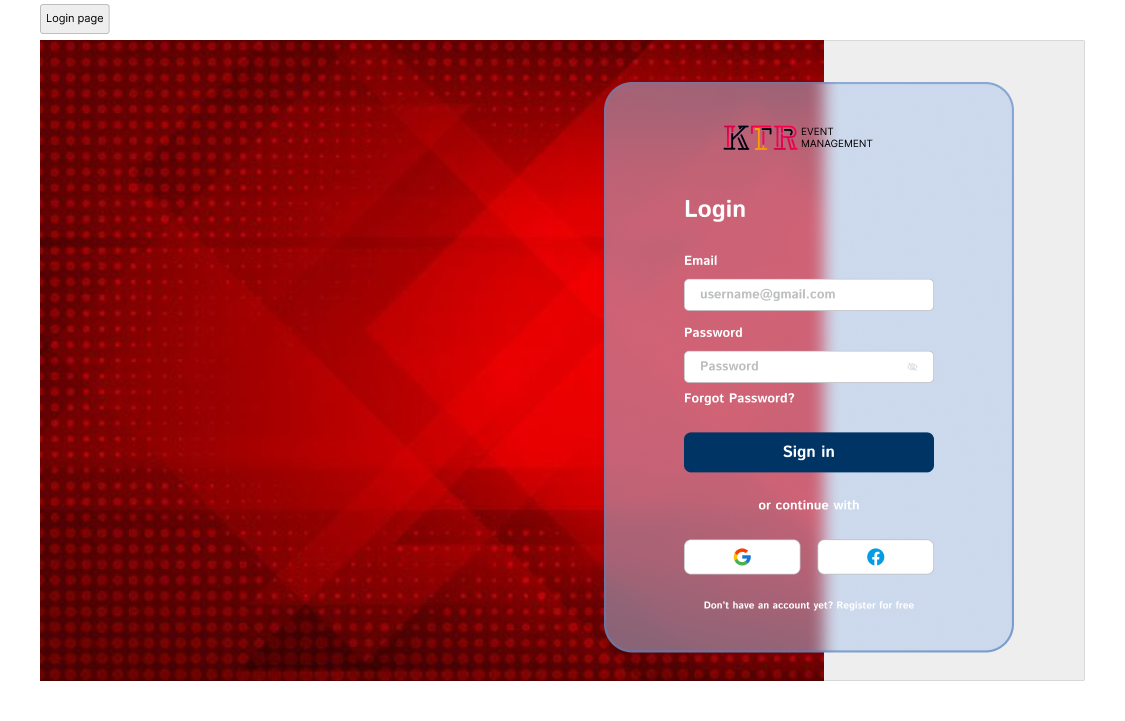
The system provides two user interfaces which is UI for User Main Page and Admin Page. Inside User Main Page is where the user searches, applies and requests for booking event places, activities and packages in KTR. While Admin Page is where the administrator approves/rejects user requests, manages event places, status, activities and packages.

## **5.2** **Screen Images**

**Registration Page**

# 

**Login Page**

****

# 

# 

# 

# 

# 

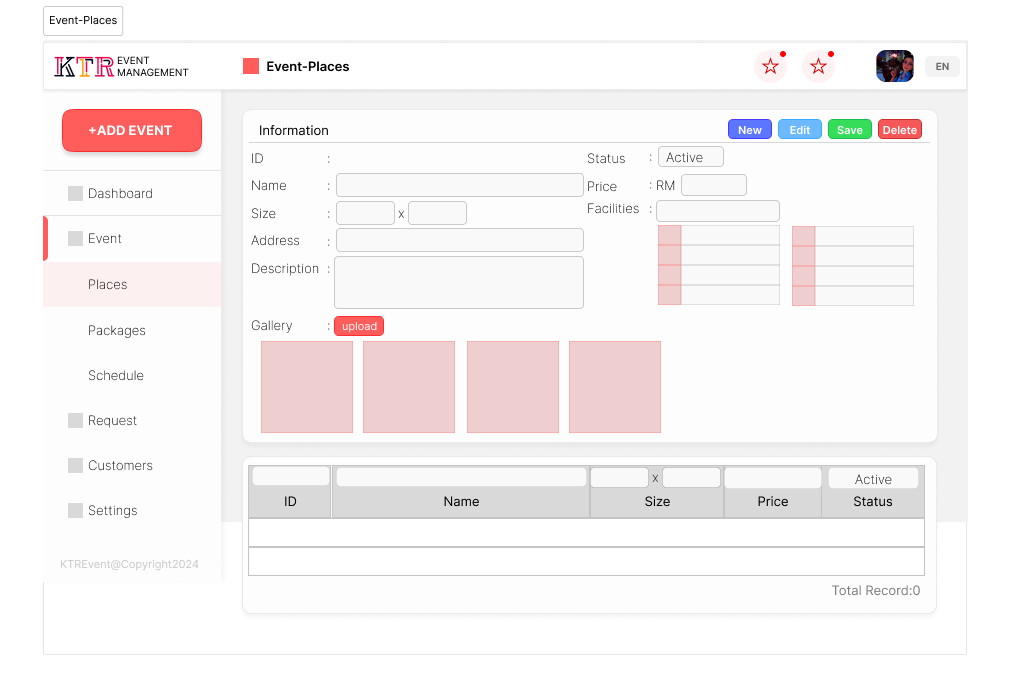
# 

# **Admin UI:**

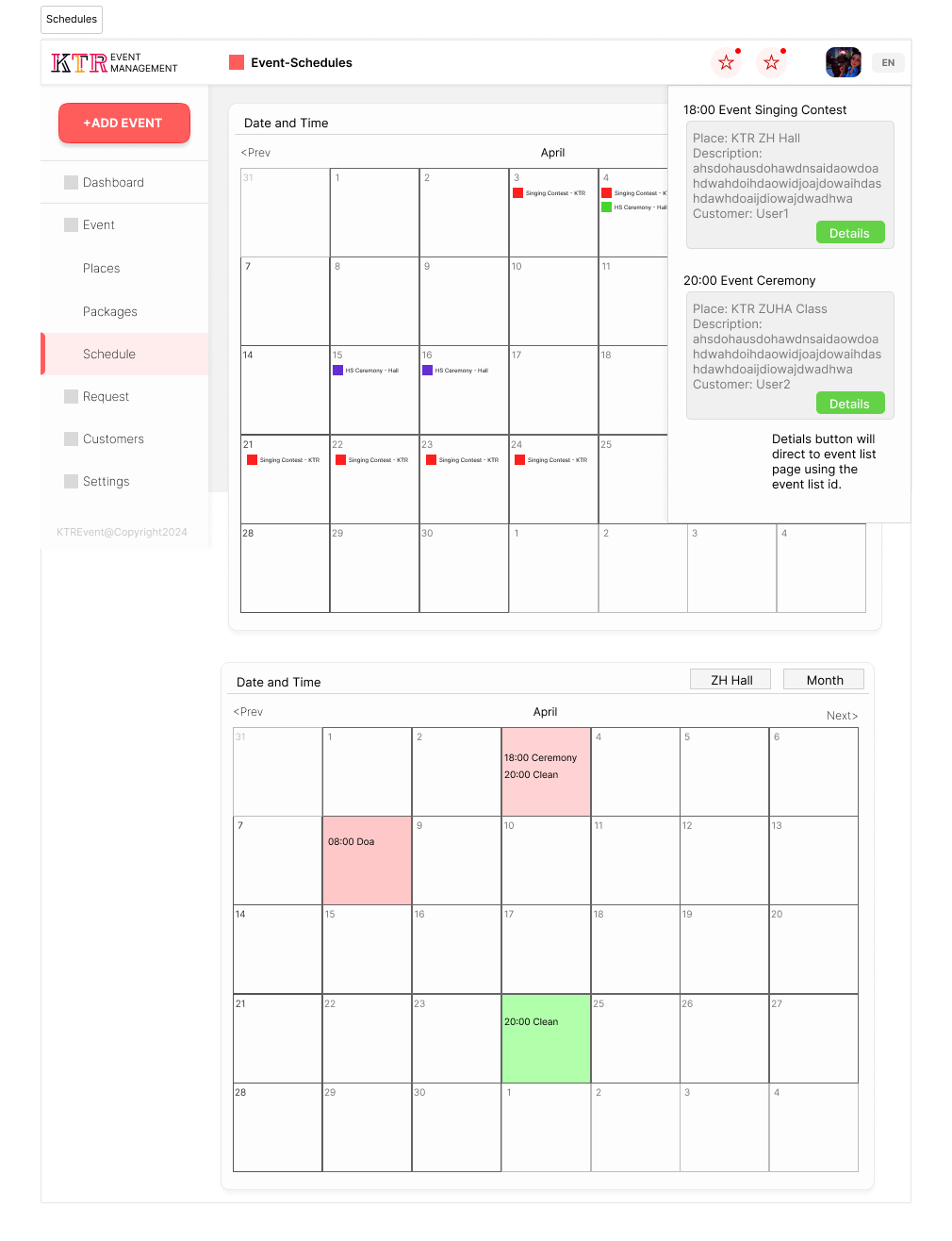
**MainPage(Admin)**

****

**Event-Places**

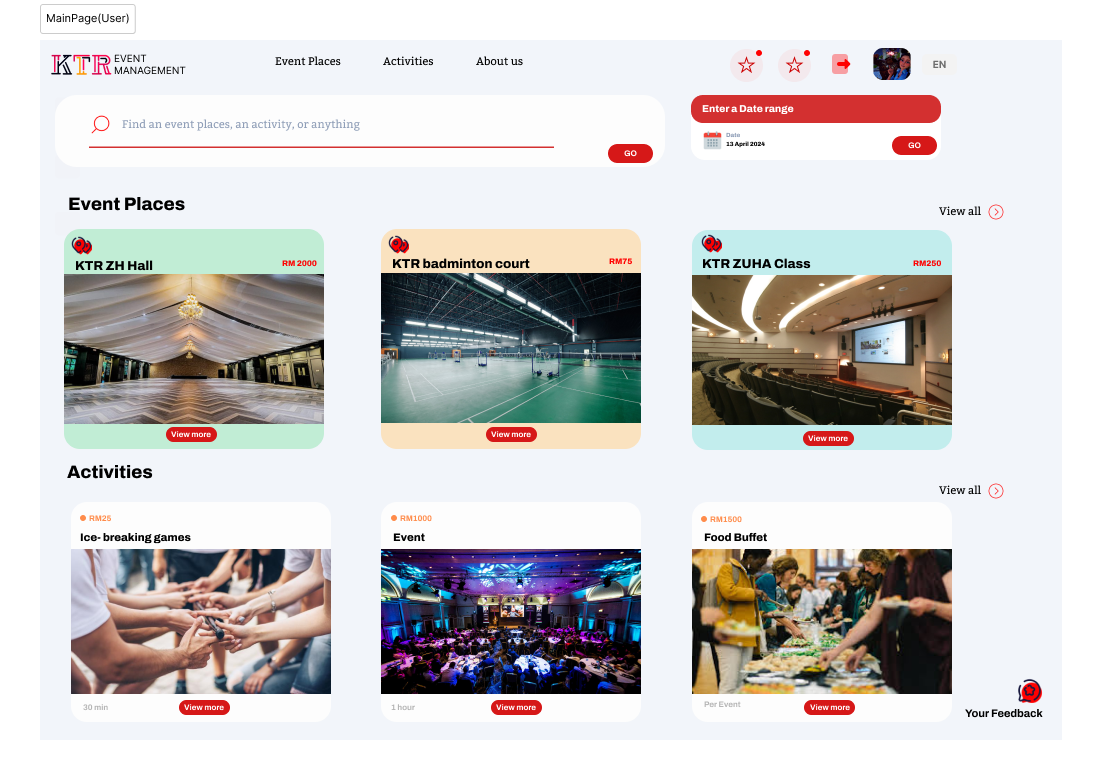
****

**Schedules**

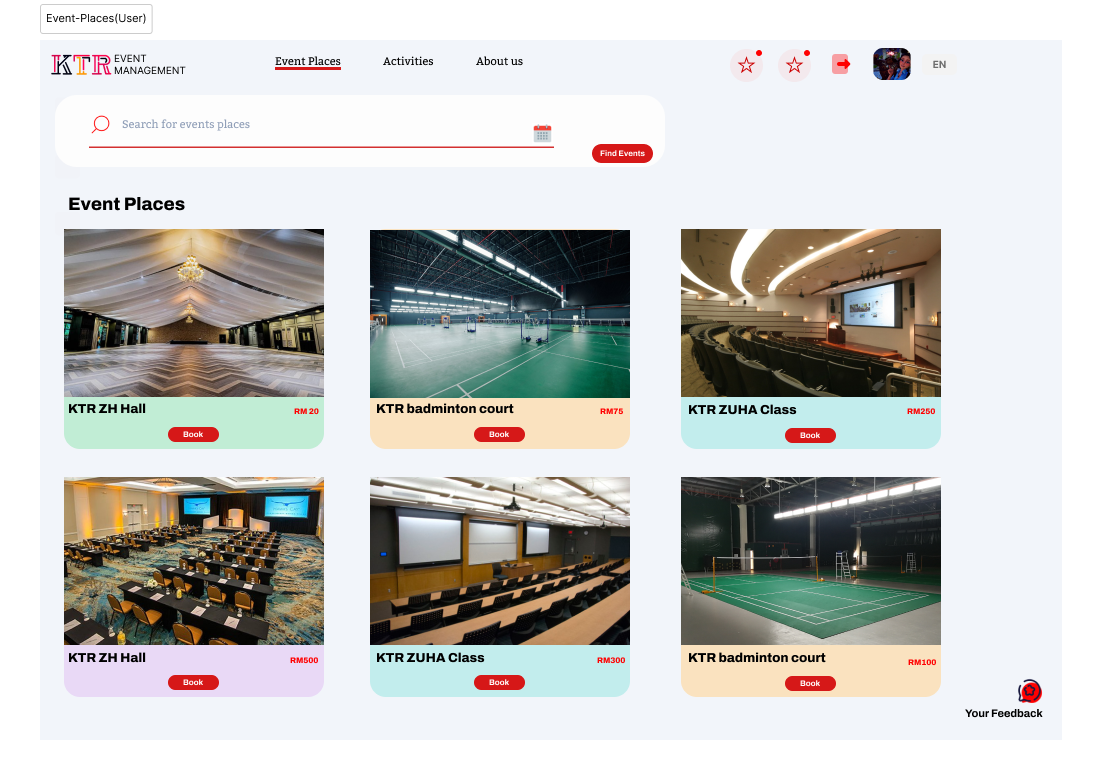
****

**User UI:**

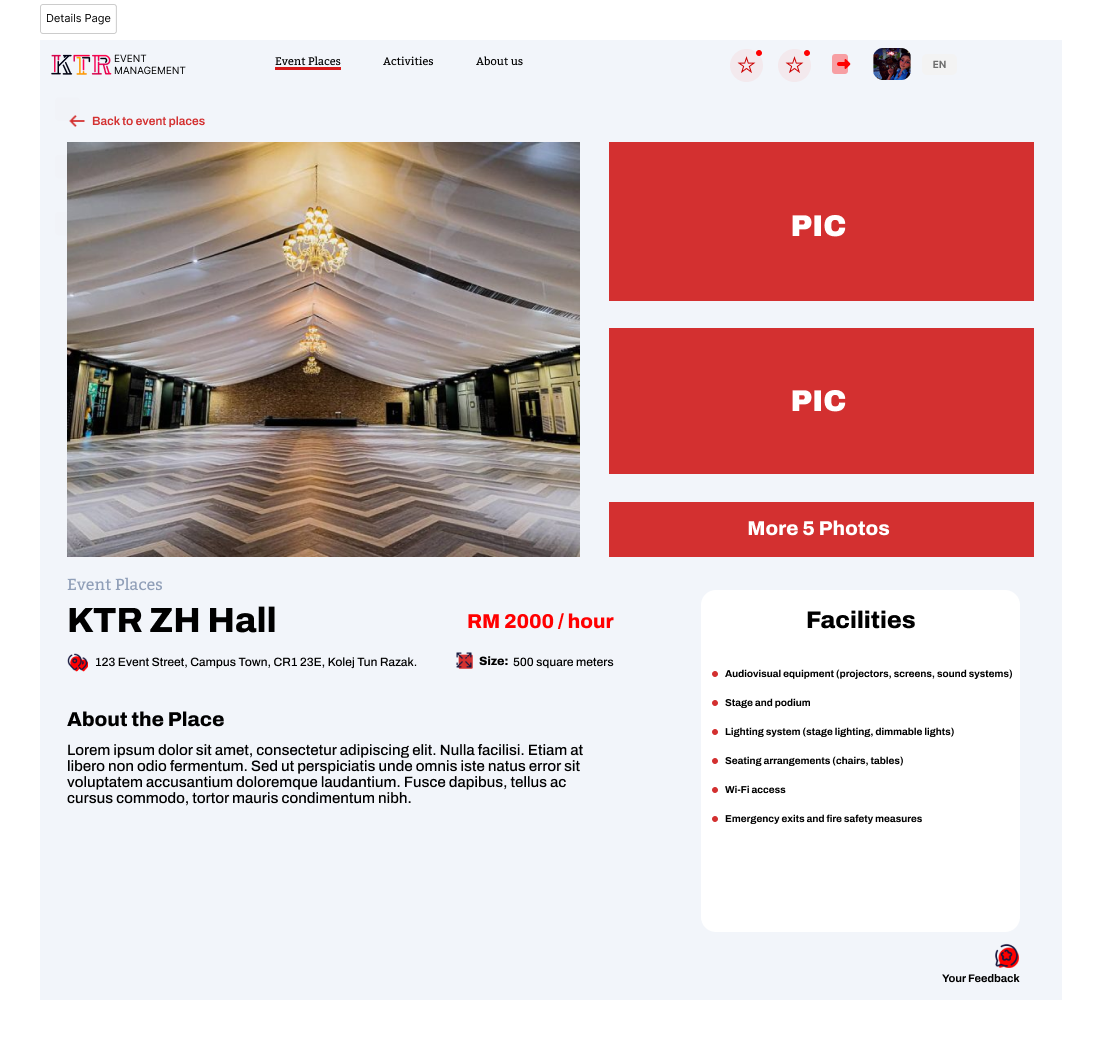
**MainPage(User)**

****

**Event-Places(User)**

****

**Details Page**

****

**About Us**

****

# **6.0 Requirements Matrix**

|  |  |  |  |
| --- | --- | --- | --- |
|  | P001 | P002 | P003 |
| UC001 | X |  |  |
| UC002 | X |  |  |
| UC002 | X |  |  |
| UC004 | X |  | X |
| UC005 | X |  | X |
| UC006 | X |  |  |
| UC007 | X |  |  |
| AC001 |  | X |  |
| AC002 |  | X |  |
| AC003 |  | X | X |
| AC004 |  | X |  |
| AC005 |  | X |  |
| AC006 |  | X |  |

# 

# **7**.**0 Appendices**