CIS016-1 – Principles of Programming – 2019-2020  
CIS096-1 – Principles of Programming and Data Structures – 2019-2020

Assignment 2- Group / Individual Project – Extra-Curricular Events Management System

DRAFT DESIGN

Functional Requirements

EEMS = Extra-Curricular Events Management System

|  |  |  |
| --- | --- | --- |
| Req. No | Requirement | Priority\* |
| 1 | As student must be able to register on the EEMS | MUST |
| 2 | A student must be able to log in to the EEMS | MUST |
| 3 | A student must be able to log out of the EEMS | COULD |
| 4 | An administrator must be able log in to the EEMS | SHOULD |
| 5 | An administrator must be able to log out of the EEMS | COULD |
| 6 | An administrator must be able to grant Organiser rights to a student | SHOULD |
| 7 | An administrator must be able to revoke Organiser rights from a student | SHOULD |
| 8 | A student must be able to search events | SHOULD |
| 9 | A student must be able to view all events | MUST |
| 10 | A student must be able to book an event | MUST |
| 11 | A student must be able to view his/her bookings | MUST |
| 12 | A student must be able to cancel a booking | MUST |
| 13 | An organiser must be able to post an event | MUST |
| 14 | An organiser must be able to view all his/her events | SHOULD |
| 15 | An organiser must be able to edit an event | SHOULD |
| 16 | An organiser must be able to cancel an event | SHOULD |
| 17 | An administrator must be able to view all events | SHOULD |
| 18 | An administrator must be able to search all events | COULD |
| 19 | An administrator must be able to cancel an event | SHOULD |
| 20 | An administrator must be able to view all bookings | SHOULD |
| 21 | An administrator must be able to cancel a booking | SHOULD |
| 22 | An administrator must be able to confirm a booking | SHOULD |

Non-functional Requirements

|  |  |  |
| --- | --- | --- |
| Req. No | Requirement | Priority\* |
| 1 | The EEMS should process input and return results within 10 seconds |  |
| 2 | The EEMS should run on a range of devices from PCs to mobile phones |  |
| 3 | The EEMS design should be sufficiently scalable and flexible to allow for further future enhancements |  |
| 4 | The EEMS users should not experience critical system failures. 99.99% ‘uptime’ should be achieved. |  |

Usability Requirements

|  |  |  |
| --- | --- | --- |
| Req. No | Requirement | Priority\* |
| 1 | The EEMS should incorporate a user-centric design |  |
| 2 | The design should demonstrate evidence of a good understanding of interface design issues – for example, a consistent design for each form, layout of content, use of colour schemes and images, navigational methods, usability when viewed at various screen resolutions and various monitor dimensions. |  |
| 3 | All data entry forms should be short and easy to complete and there should be entry validation. |  |
| 4 | The EEMS short have clear and intuitive navigation |  |
| 5 | The EEMS should comply with WW3 Web Accessibility Standards (WCAG)  Text easy to read and language and language style should be appropriate with absence of grammar / spelling errors |  |
|  | There should be a clear layout which remains consistent throughout the application. Style, layout and content should be appropriate for the purpose of the application. |  |

\*MOSCOW Notation:

M = MUST

S = SHOULD

C = COULD

W = WON’T

Design

UML Diagrams

Use Case Diagram

A close up of a map

Description automatically generated

Activity Diagram

Class Diagram

A screenshot of a computer

Description automatically generated

Classes

HIDE THIS SECTION

Student (studentid, universityid, password, firstname, lastname, email, role)

Event (eventid, eventtitle, eventdescription, category, date, time, location, roomno, places, bookingrequired)

Booking (bookingid, datebooked, status)

Administrator (adminid, universityid, password, firstname, lastname, email)

Connection (connectionid, hostname, database, username, password)

Database Design

VERSION 1

Entity Relationship Model (ERM)

Entity Relationship Diagram (ERD)

Admin

1

0.1.M

0.1.M

1

1

makes

is for

has

is for

Event

Booking

Student

is for

posts

is organized by

Connection

is organized by

0.1.M

0.1.M

Event Organiser

A picture containing screenshot, computer

Description automatically generated

Entities

Student (**studentid**, universityid, password, firstname, lastname, email, role)

Event (**eventid**, eventtitle, eventdescription, category, date, time, location, roomno, places, bookingrequired)

Booking (**bookingid**, datebooked, status)

EventOrganiser (**studentid, eventid**)

Administrator (**adminid**, universityid, password, firstname, lastname, email)

Connection (**connectionid**, hostname, database, username, password)

Skeleton Tables

Student (**studentid**, universityid, password, firstname, lastname, email, role)

Event (**eventid**, eventtitle, eventdescription, category, date, time, location, roomno, places, bookingrequired)

Booking (**bookingid**, studentid\*, eventid\*, datebooked, status)

EventOrganiser (**studentid\*, eventid\***)

Administrator (**adminid**, universityid, password, firstname, lastname, email)

Connection (**connectionid**, hostname, database, username, password)

Primary Key = Bold and Underlined

Foreign Key = \*

VERSION 2 (SIMPLER)

Entity Relationship Model (ERM)

Entity Relationship Diagram (ERD)

Admin

0.1.M

0.1.M

1

1

makes

is for

has

is for

Event

Booking

Student

1

0.1.M

is organized by

organises

Connection

A picture containing screenshot

Description automatically generated

Entities

Student (**studentid**, universityid, password, firstname, lastname, email, role)

Event (**eventid**, eventtitle, eventdescription, category, date, time, location, roomno, places, bookingrequired)

Booking (**bookingid**, datebooked, status)

Administrator (**adminid**, universityid, password, firstname, lastname, email)

Connection (**connectionid**, hostname, database, username, password)

Skeleton Tables

Student (**studentid**, universityid, password, firstname, lastname, email, role)

Event (**eventid**, studentid\*, eventtitle, eventdescription, category, date, time, location, roomno, places, bookingrequired)

Booking (**bookingid**, studentid\*, eventid\*, datebooked, status)

Administrator (**adminid**, universityid, password, firstname, lastname, email)

Connection (**connectionid**, hostname, database, username, password)

Primary Key = Bold and Underlined

Foreign Key = \*

Physical Database Design

Data Dictionary

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Student | | | | | | | |
| Description: Student details | | | | | | | |
| Field Name | **Datatype** | **Length** | **Index** | **Null** | **Default** | **Validation rule** | **Description** |
| studenttid *(Primary)* | int (11)  unsigned | 11 | PK | No |  |  | Autoincremented Uniquely identifies every student |
| universityid | varchar (10) | 10 |  | No |  |  | University Id od student |
| password | varchar (30) | 30 |  | No |  |  | EEMS password |
| firstname | varchar (30) | 30 |  | No |  |  | First name of customer |
| lastname | varchar (30) | 30 |  | No |  |  | Last name of customer |
| email | varchar (100) | 100 |  | No |  | Must be email format containing an @ and a ‘.’  Regex expression used | Email of student |
| role | varchar (10) | 10 |  | No |  |  | Student or Organiser |

Indexes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Column** | **Null** |
| PRIMARY | BTREE | Yes | custid | No |

User Interface Design (UI)

…….