## **ASSIGNMENT-3 DESIGN OVERVIEW**

### **Entities and Their Responsibilities**

1. **Event**

* date: The date of the event (e.g., "2024-10-01").
* start: The start time of the event (e.g., "12:00").
* ends: The end time of the event (e.g., "15:00").
* eventname: The name of the event (e.g., "Javelin Throw").

The Event class encapsulates the details of an event, including its date, start and end times, and name. It is used within the Venue class to manage events occurring at a particular venue.

1. **Congregation**

* name: The name of the congregation (e.g., "Paris Olympics 2024").
* type: The type of congregation (e.g., "Games").
* startdate: The start date of the congregation.
* enddate: The end date of the congregation.

The Congregation class represents a gathering or event type with a specific duration. It is linked to venues by congregation name and is managed by the CongregationManager.

1. **Venue**

* name: The name of the venue (e.g., "Paris Stadium").
* location: The location string in the format "City:State:PostalCode
* country: The country where the venue is located.
* city: The city where the venue is located.
* state: The state where the venue is located.
* postalCode: The postal code of the venue's location.
* capacity: The seating capacity of the venue.
* congregationName: The name of the congregation currently reserving the venue.
* events: An array of Event objects associated with the venue.
* eventCount: The count of events scheduled at the venue.
* isReserved: A flag indicating if the venue is currently reserved.
* **Methods:**
* Venue(): Default constructor.
* Venue(const string &n, string &loc, int cap): Parameterized constructor to initialize a venue.
* addEvent(string event, string date, string start, string end): Adds an event to the venue.
* deleteEvent(const string &eventname, string date, string start): Deletes an event from the venue.
* showEvents(string date) const: Displays events scheduled on a specific date.
* showCalendar(string start, string end) const: Shows a calendar of events between two dates.

The Venue class manages information about a venue, including its location, capacity, and events. It maintains a list of events scheduled at the venue and handles operations related to event management.

1. **VenueManager**

* addVenue(const string &name, const string &location, int capacity): Adds a new venue to the system.
* deleteVenue(const string &name, const string &country): Deletes a venue from the system.
* showVenues(const string &location): Shows venues matching a specified location.
* addEvent(const std::string &Name, const std::string &venueName, const std::string &country, const std::string &date, const std::string &start, const std::string &end, const std::string &eventname): Adds an event to a specified venue.
* deleteEvent(const string &Name, const string &venueName, string country, string date, string fromtime, const string &eventname): Deletes an event from a specified venue.
* showEvents(const string &venueName, const string &country, string date) const: Displays events at a specific venue on a given date.
* showCalendar(const string &name, const string &venueName, string country) const: Shows the calendar of events for a specific congregation at a venue.
* reserveVenue(const string &venueName, const string &venueCountry, const string &congregationName): Reserves a venue for a specific congregation.
* freeVenue(const string &venueName, const string &venueCountry, const string &congregationName): Frees a venue from reservation.
* showReserved(const string &congregationName) const: Displays venues reserved for a specific congregation.

The VenueManager class provides operations for managing venues, including adding, deleting, and reserving venues, as well as managing events associated with those venues.

1. **CongregationManager**

* int addCongregation(const string &name, const string &type, const string &startDate, const string &endDate): Adds a new congregation.
* int deleteCongregation(const string &name): Deletes a congregation from the system.
* void showCongregations() const: Displays all congregations.

The CongregationManager class handles the creation and deletion of congregations and displays the list of existing congregations.

## **Relationships**

* **VenueManager** has a one-to-many relationship with **Venue**, indicating that one manager can handle multiple venues.
* **CongregationManager** has a one-to-many relationship with **Congregation**, indicating that one manager can handle multiple congregations.
* **Venue** has a one-to-many relationship with **Event**, indicating that one venue can host multiple events.
* **Venue** has a one-to-one relationship with **Congregation**, indicating that a venue can be associated with one congregation at a time.

## **Design Considerations**

* **Encapsulation:** Each class encapsulates its attributes and provides methods to manipulate these attributes. For example, Venue handles events internally and provides methods to add or delete events, ensuring that the internal state remains consistent.
* **Modularity:** The system is designed to be modular, with clear separation of responsibilities among different classes. VenueManager handles venue-related operations, while CongregationManager handles congregations.
* **Reusability:** The Event class can be reused across different venues, promoting code reuse and consistency.

