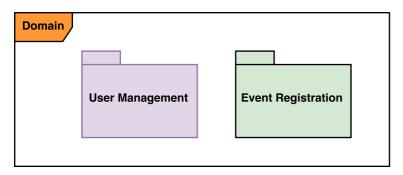
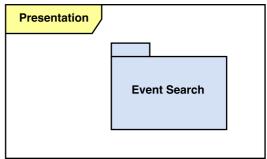
Design Model

Assignment 2

Package diagram





Design model

EventDataAccess

- + queryEvents(filter: EventFilter) : List<Event>
- + getRegisteredEvents(user: User) : List<Event>
- + registerEvent(event: Event) : void

PerformerController

+ registerPerformer(performer: Performer) : void

EventRegistrationController

- + getRegisteredEvents(user: User) : List<Event>
- + registerEvent(event: Event) : void
- + changeEvent(event: Event) : void

VenueController

+ registerVenue(venue: Venue): void

Event

- title: String - time : Date
- eventTypeld: int
 venue: Venue
- performers: List<performer> eventOwners: List<User>
- + getEventType(): String
- + getVenue(): String
- + getPerformer(): String
- + getTitle(): String
- + setTime(time: Date) : void
- + setEventType(type: int): void
- + setVenue(venue: Venue): void
- + setTitle(title: String): void
- + setPerformer(performer: Performer)

(0..n

Venue

(o..n

+ addEventOwner(user: User) : void

EventTypes

- CONCERT TYPEID: int
- CONCERT_TYPE_DESCRIPTION: String
- STANDUP_TYPEID: int
- STANDUP_TYPE_DESCRIPTION: String
- LECTURE_TYPEID: int
- LECTURE_TYPE_DESCRIPTION : String
- ART_EXHIBIT_TYPEID: int
- ART_EXHIBIT_DESCRIPTION : String
- DANCE TYPEID: int
- DANCE_DESCRIPTION: String
- + getDescription(typeId:int) : String
- + getConcertType(): int
- + getStandUpType() : int + getLectureType() : int
- + getArtExhibitType() : int
- + getDanceType(): int

EventQueryController

+ queryEvents(filter: EventFilter) : List<Event>

EventFilter

- name : String
- performerId: int
- startDate: Date
 endDate: Date
- + getName() : String
- + setName(name : String) : void
- + getPerformerId() : int
- + setPerformerId(performerId: int) : void
- + getStartDate(): Date
- + setStartDate(date : Date) : void
- + getEndDate() : Date
- + setEndDate(date : Date) : void

UserAuthentificationController

- + authenticateUser(user: User) : boolean
- + createVenueManager(venueManager:
- VenueManager) : boolean
- + createPerformerManager(performerManager:

PerformerManager) : boolean

UserDataAccess

- + authenticateUser(user: User) : boolean
- + createVenueManager(venueManager:
- VenueManager) : boolean
- + createPerformerManager(performerManager:

PerformerManager): boolean

User

- userName: String
 passwordSalt: String
- passwordHash: String
 isAdmin: boolean
- + getUsername(): String
- + setUsername(userName: String): void
- + getPasswordSalt(): String
- + setPasswordSalt(salt: String): void
- + getPasswordHash() : String
- + setPasswordHash(hash: String): void
- + isAdmin() : boolean
- + setAdmin(isAdmin: boolean): void

+ getName(): String

- name: String

address: String
 city: String

+ getAddress(): String

hasElevator: boolean

- + getCity(): String
- + isWheelchairAccessible(): boolean

isWheelchairAccessible: boolean

- + hasElevator(): boolean
- + setAccessibility(isAccessible: boolean): void
- + setElevatorAccess(hasElevator : boolean): void
- + setName(name: String): void
- + setAddress(address: String): void
- + setCity(city: String): void

VenueManager

- venue : Venue
- + getVenue() : Venue
- + setVenue(venue : Venue) : void

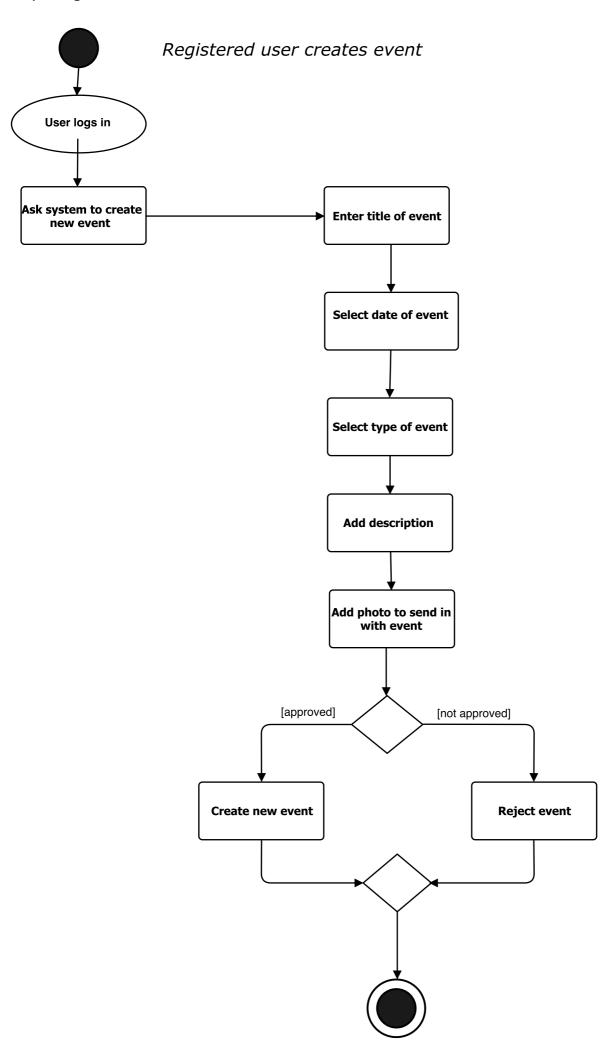
PerformerManager

- performer : Performer
- + getPerformer() : Performer + setPerformer(performer : Performer) : void

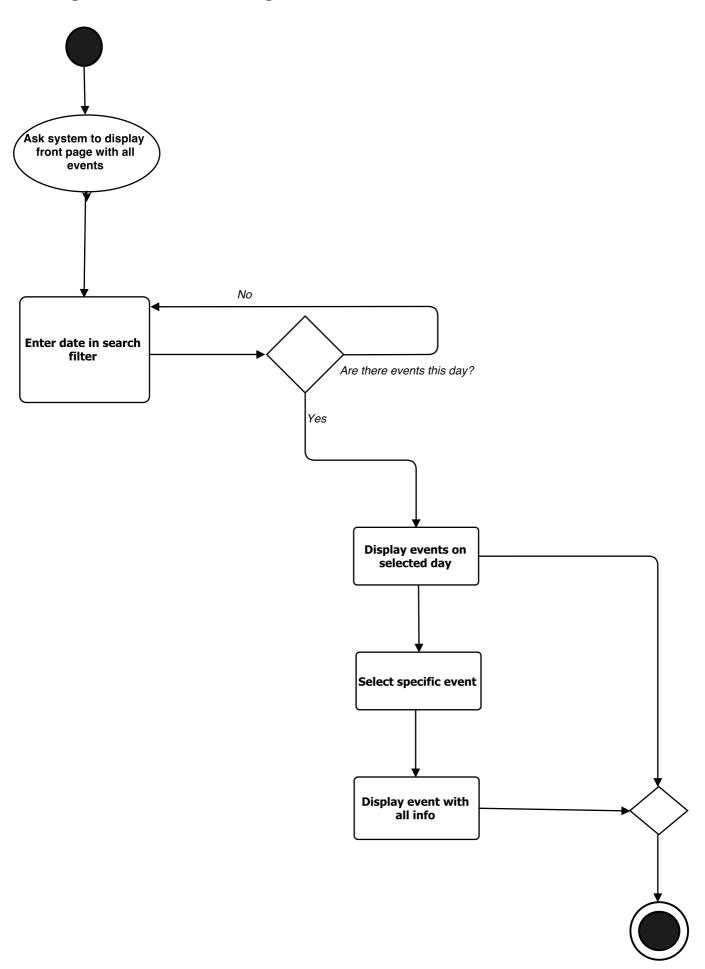
- name: String - type: String - description : String

- + getName(): String
- + setName(name: String): void
- + getType(): String
- + setType(type: int): void
- + getDescription() : String
- + setDescription(description: String): void

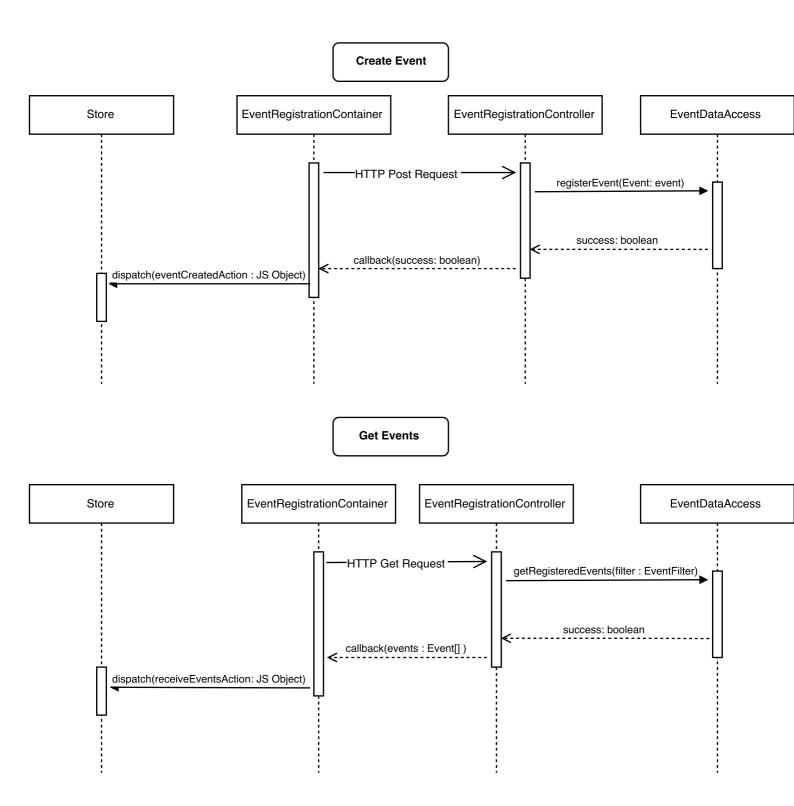
Performer



Searching for event as an unregistered user



Sequence Diagram



State Diagram

