Class Diagram

Authenticatable
- email: String
+ name: String
- password: String

+ Authenticatable authenticate (username: String, password: String)

BusinessOwner
- phone_number: Long
+ city: String

+ Search_Place(name: String, city: String = city): Place[]
+ Search_Event(name: String, city: String = city, start_time: Datetime.time, end_time: Datetime.time, type: String, full: Bool = None): Event[]
+ Post_Event(name: String, start_time: datetime.time, end_time: datetime.time, description: String, max_ppl: Int, current_ppl: Int, price: Int, type: String): Event
+ Delete_Event(e: Event): void

Client
- phone_number: Long
+ city: String
+ follows: Client[]

+ Search_Place(name: String, city: String = city): Place[]
+ Search_Event(name: String, city: String = city, start_time: Datetime.time, end_time: Datetime.time, type: String, full: Bool = None): Event[]
+ RSVP_Event(e: Event): void
+ Cancel_Event(e: Event): void
+ View_Map(c: Client): Place[]
+ Add_To_Map(p: Place): void
+ Delete_From_Map(p: Place): void
+ Post_Event_Review(e: Event, content: String, rating: Int) : EventReview
+ Delete_Event_Review(er: EventReview): Void
+ Get_Followers(): Client
+ Get_Follows(): Client
+ Follow(c: Client)
+ Unfollow(c: Client)

follow

post

RSVP
+ id: Int
+ event: Event
+ client: Client

Place
- id: Int
+ name: String
+ location_altitude: Double
+ location_latitude: Double
+ city: String

+ Get_Name(): String
+ Get_location_latitude(): Double
+ Get_location_altitude(): Double
+ Get_city(): String
+ Set_Name(name: String): Void

Map
+ id: Int
+ place: Place[]
+ client: Client

Take Place

Event
- id: Int
+ name: String
+ place: Place
+ business_owner: BusinessOwner
+ start_time: Datetime.time
+ end_time: Datetime.time
+ description: String
+ max_ppl: Int
+ current_ppl: Int
+ score: Short
+ avg_price: float
+ type: String

+ Get_Name(): String
+ Get_Place(): Place
+ Get_Business_Owner(): BusinessOwner
+ Get_Start_Time(): Datetime.time
+ Get_End_Time(): Datetime.time
+ Get_Description(): String
+ Get_Max_Ppl(): Int
+ Get_Current_Ppl(): Int
+ Get_Score(): Int
+ Get_Type(): String
+ Get_Name(): String
+ Get_Place(): Place
+ Get_avg_price(): float
+ Set_Business_Owner(BusinessOwner): Void
+ Set_Start_Time(Datetime.time): Void
+ Set_End_Time(Datetime.time): Void
+ Set_Description(String): Void
+ Set_Max_Ppl(Int): Void
+ Set_Type(String): Void
+ Update_Current_Ppl(ppl: Int): Void
+ Update_score_and_spending(er: EventReview): Void
+ Check_Full(): Bool

EventReview
+ id: Int
+ event: Event
+ client: Client
+ content: String
+ rating: Int

+ Get_content(): String
+ Get_rating(): Int
+ Set_content(content: String): void
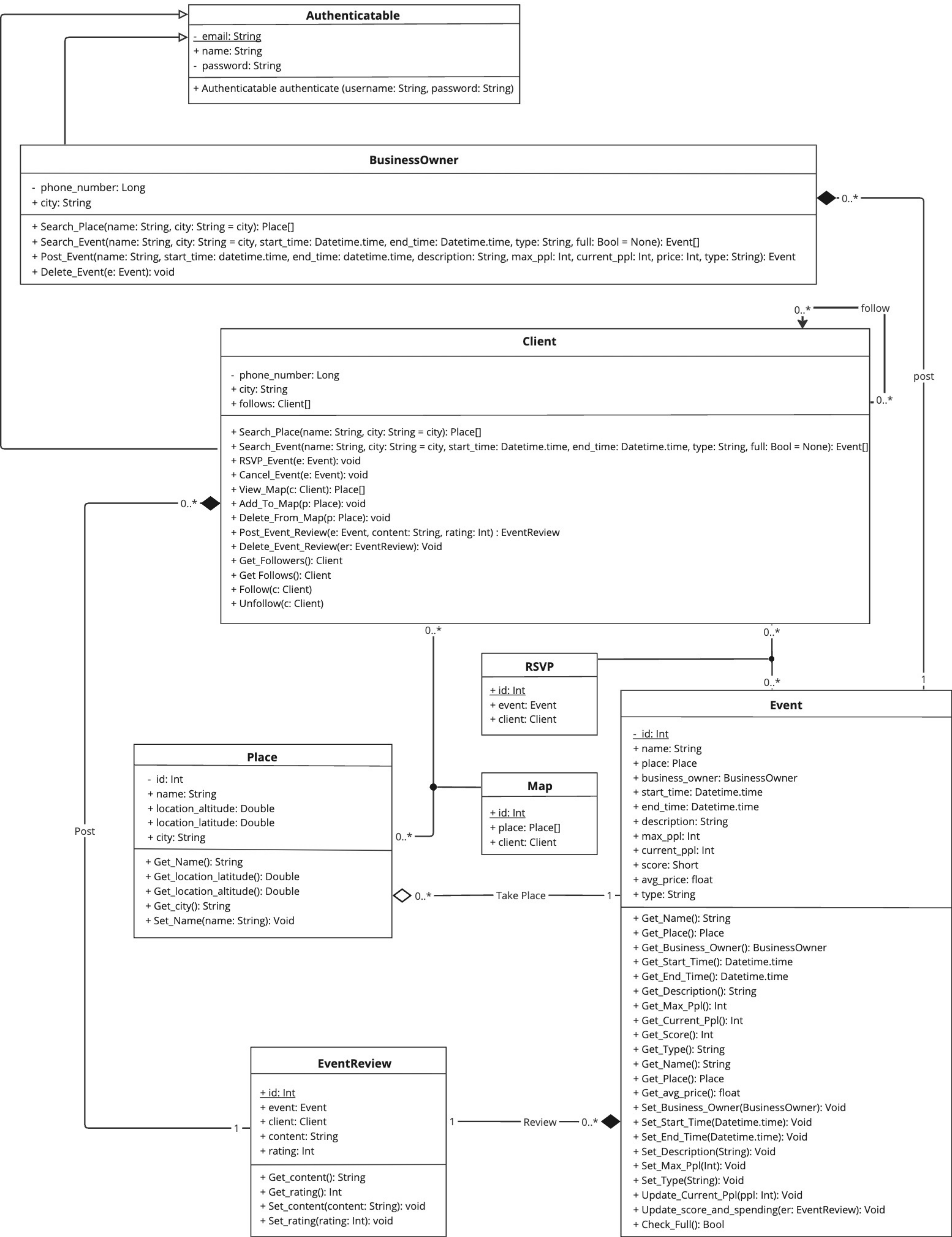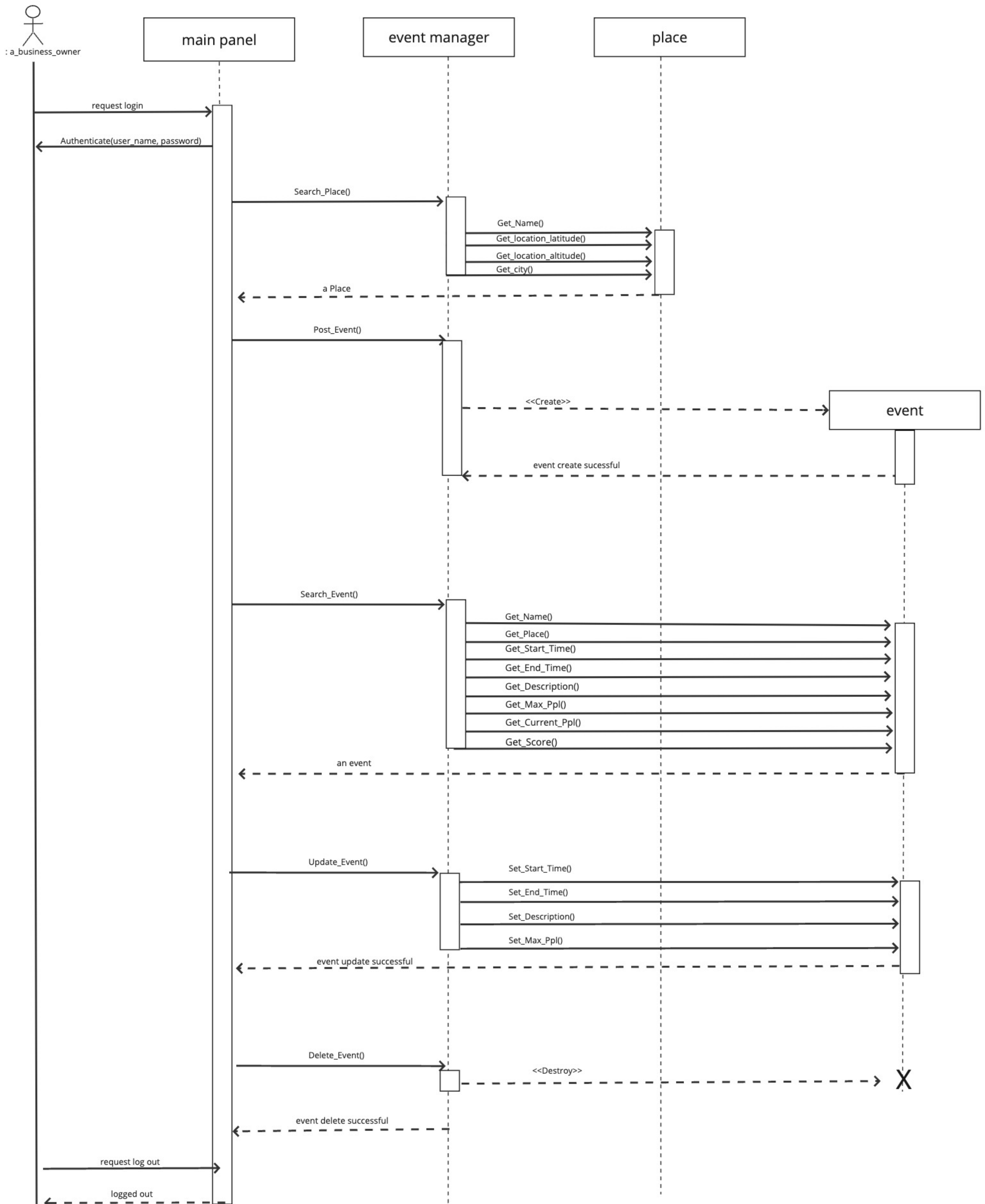+ Set_rating(rating: Int): void

Post

Review

Sequence Diagram 1(Shuochen Zhao, Mei Han)

User Story of a business owner:

   As a business owner, I want to be able to search for a desirable place, and then post a new event on the app, so that I can advertise my upcoming events to potential clients. Additionally, I want to access my owned event via event search. Furthermore, when there is anything that I would like to change or update about an event, I should be able to modify it. Finally, when I want to close an event, I should be able to delete the events that have been posted.

Sequence Diagram 2(Ziye Jin, Shiyuan Tian)

User Story of a client:

As a client, I have various functionalities: search for places by name and city; search for events by name, city, start time, end time, and type; RSVP to events to indicate my attendance; cancel my attendance at an event if I change my mind; view a map of places I am interested in; add new places to my map; delete places from my map; post reviews for events I have attended; delete my reviews if I choose to; follow other clients to connect with them; unfollow clients if I no longer wish to be connected; view a list of clients who follow me and a list of clients I am following.