FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Event Planner - Copy

Design

Documentation of a project for the purpose of the course BIE-SI1.

Authors:

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Contents

1. Architecture

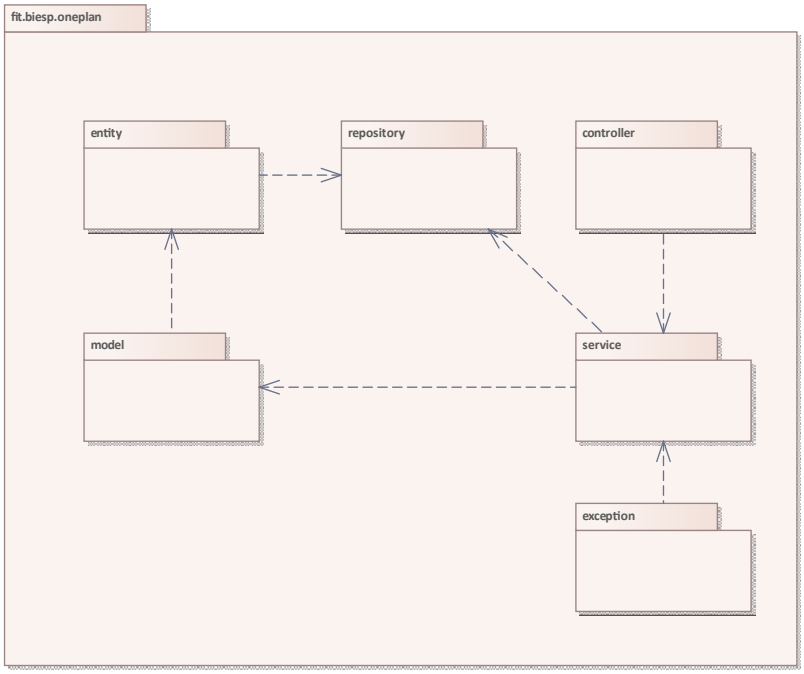

Figure 1 - Architecture

Architecture defines the boundaries between parts of the app and the responsibilities each part should have. There are several data classes, entity is used to store information in the database, while DAO classes are used to transfer information between user and the application. HTTP requests are handled by controller classes. Service classes are used for logic of the app and data manipulations. Repository interfaces extend CrudRepository and used for connection with a database. Exceptions are handled by exception classes.

1.1 app

1.2 controller

Used for HTTP requests handlling.

1.3 entity

Entities are used to store information in a database.

Figure 2 - entity

1.4 exception

Used for exceptions that might occur.

Figure 3 - exception

1.5 fit.biesp.oneplan

Figure 4 - fit.biesp.oneplan

1.6 model

Model classes used to transfer information between user and the application

Figure 5 - model

1.7 repository

Used for connection with a database.

Figure 6 - repository

1.8 security

Used for secure user log in and registration.

Figure 7 - security

1.9 service

Implementation of app logic and data manipulations.

Figure 8 - service

1.10 web

2. Design Model

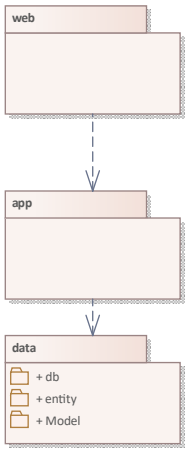

Figure 9 - Design Model

2.1 app

Figure 10 - app

2.2 data

This package contains the classes and packages of the data layer.

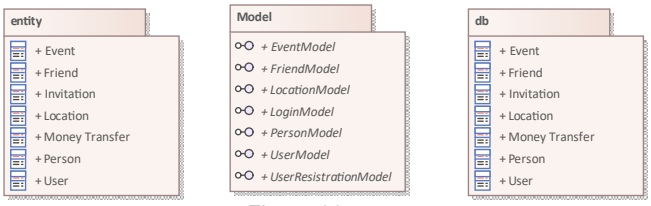

Figure 11 - data

This diagram shows the classes and packages of the data layer.

2.2.1 db

This package contains implementation of the data persistence using the PostgreSQL database storage.

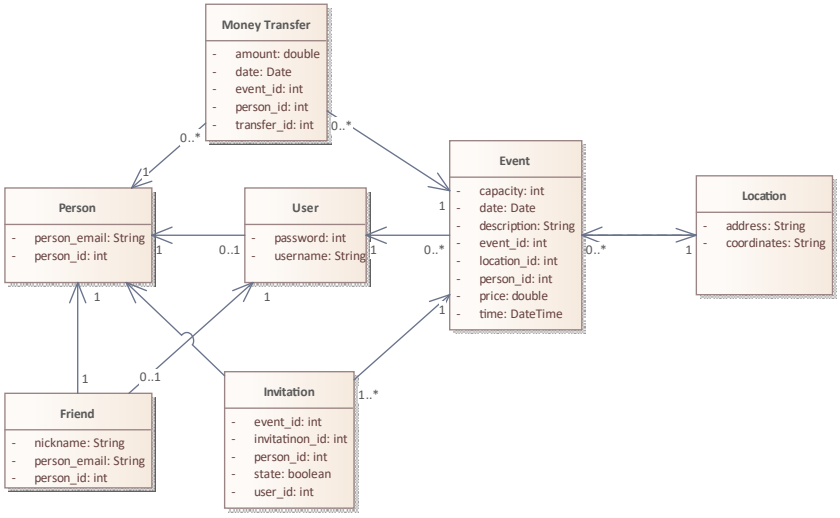

Figure 12 - db

This diagram shows the classes of the entities, defining the data objects synchronized with a persistent storage.

2.2.1.1 Event

The EVENT entity serves as a data holder for the data about event stored in a persistent storage.
Event represents a definition of an event and its details.
Event defines its capacity , date , description , unique event_id , time and price.

Attribute name	Data type	Description
capacity	int	
date	Date	
description	String	
event_id	int	
location_id	int	
person_id	int	
price	double	
time	DateTime	

2.2.1.2 Friend

The Friend entity serves as a data holder for the data about a friend stored in a persistent storage.

Attribute name	Data type	Description
nickname	String	
person_email	String	
person_id	int	

2.2.1.3 Invitation

The INVITATION entity serves as a data holder for the data about invitations to the event stored in a persistent storage.

Attribute name	Data type	Description
event_id	int	
invitatinon_id	int	
person_id	int	
state	boolean	
user_id	int	

2.2.1.4 Location

The LOCATION entity serves as a data holder for the data about location of the event stored in a persistent storage.

Attribute name	Data type	Description
address	String	
coordinates	String	

2.2.1.5 Money Transfer

The MONEY TRANSFER entity serves as a data holder for the data about money transfers stored in a persistent storage.

Attribute name	Data type	Description
amount	double	
date	Date	
event_id	int	
person_id	int	
transfer_id	int	

2.2.1.6 Person

The Person entity serves as a data holder for the data about person stored in a persistent storage.
Person represents information about a person which may be an user or guest attendee.

Attribute name	Data type	Description
person_email	String	
person_id	int	

2.2.1.7 User

The User entity serves as a data holder for the data about user stored in a persistent storage.
User is a person that can participate in creation of an event by creating it or organizing it.
A single user can create many events and participate in many others.

Attribute name	Data type	Description
password	int	
username	String	

2.2.2 entity

This package contains the classes of the entities, defining the data objects synchronized with a persistent storage.

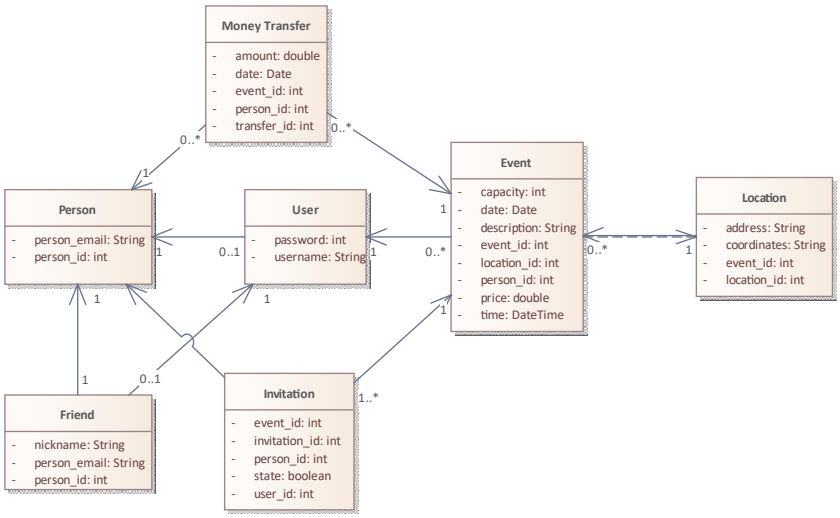

Figure 13 - entity

This diagram shows the classes of the entities, defining the data objects synchronized with a persistent storage.

2.2.2.1 Event

The EVENT entity serves as a data holder for the data about event stored in a persistent storage.
Event represents a definition of an event and its details.
Event defines its capacity , date , description , unique event_id , time and price.

Attribute name	Data type	Description
capacity	int	
date	Date	
description	String	
event_id	int	
location_id	int	
person_id	int	
price	double	
time	DateTime	

2.2.2.2 Friend

The Friend entity serves as a data holder for the data about a friend stored in a persistent storage.

Attribute name	Data type	Description
nickname	String	
person_email	String	
person_id	int	

2.2.2.3 Invitation

The INVITATION entity serves as a data holder for the data about invitations to the event stored in a persistent storage.

Attribute name	Data type	Description
event_id	int	
invitation_id	int	
person_id	int	
state	boolean	
user_id	int	

2.2.2.4 Location

The LOCATION entity serves as a data holder for the data about location of the event stored in a persistent storage.

Attribute name	Data type	Description
address	String	
coordinates	String	
event_id	int	
location_id	int	

2.2.2.5 Money Transfer

The MONEY TRANSFER entity serves as a data holder for the data about money transfers stored in a persistent storage.

Attribute name	Data type	Description
amount	double	
date	Date	
event_id	int	
person_id	int	
transfer_id	int	

2.2.2.6 Person

The Person entity serves as a data holder for the data about person stored in a persistent storage.
Person represents information about a person which may be an user or guest attendee.

Attribute name	Data type	Description
person_email	String	
person_id	int	

2.2.2.7 User

The User entity serves as a data holder for the data about user stored in a persistent storage.
User is a person that can participate in creation of an event by creating it or organizing it.
A single user can create many events and participate in many others.

Attribute name	Data type	Description
password	int	
username	String	

2.2.3 Model

This package defines the interfaces for DAOs - Data access objects - responsible for retrieving data from the persistent storage and persisting the changes. It maps the database tables to entities of the system.

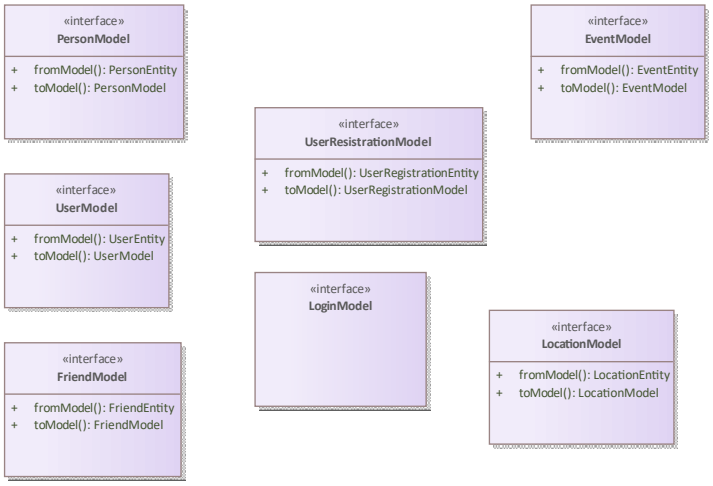

Figure 14 - model

This diagram shows the interfaces for DAOs.

2.2.3.1 EventModel

Interface defining the operations available for persistence of events

Method name	Return type	Description
fromModel	EventEntity	Returns EventEntity from EventModel.
toModel	EventModel	Returns EventModel from EventEntity.

2.2.3.2 FriendModel

Interface defining the operations available for persistence of a friend

Method name	Return type	Description
fromModel	FriendEntity	FriendModel --> FriendEntity.
toModel	FriendModel	FriendEntity --> FriendModel.

2.2.3.3 LocationModel

Interface defining the operations available for persistence of locations

Method name	Return type	Description
fromModel	LocationEntity	LocationModel --> LocationEntity.
toModel	LocationModel	LocationEntity --> LocaionModel.

2.2.3.4 LoginModel

Interface defining the operations available for persistence of login of a user

2.2.3.5 PersonModel

Interface defining the operations available for persistence of a person.

Method name	Return type	Description
fromModel	PersonEntity	PersonModel --> PersonEntity.
toModel	PersonModel	PersonEntity --> PersonModel.

2.2.3.6 UserModel

Interface defining the operations available for persistence of a user

Method name	Return type	Description
fromModel	UserEntity	UserModel --> UserEntity.
toModel	UserModel	UserEntity --> UserModel.

2.2.3.7 UserResistrationModel

Interface defining the operations available for persistence of registration of a user

Method name	Return type	Description
fromModel	UserRegistrationEntity	UserRegistrationModel --> UserEntity.
toModel	UserRegistrationModel	UserEntity --> UserRegistrationModel.

2.3 web

The web package contains classes implementing the web user interface.

Figure 15 - web

3. Realization Model

An implementation of a given input-output behavior.
The realization model describes the realization of important Use Cases in the logic and structure of the system. Such realization is usually described by sequence diagrams consisting of lifelines of various objects and method calls between them.
In this chapter, the realization of some of the use cases of the EVENT PLANNER is described, showing the communication of classes of the Web application

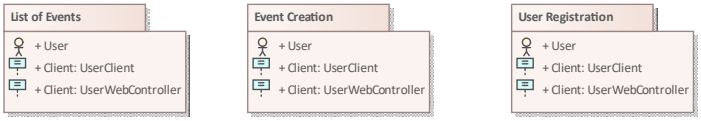

Figure 16 - Realization Model

3.1 Event Creation

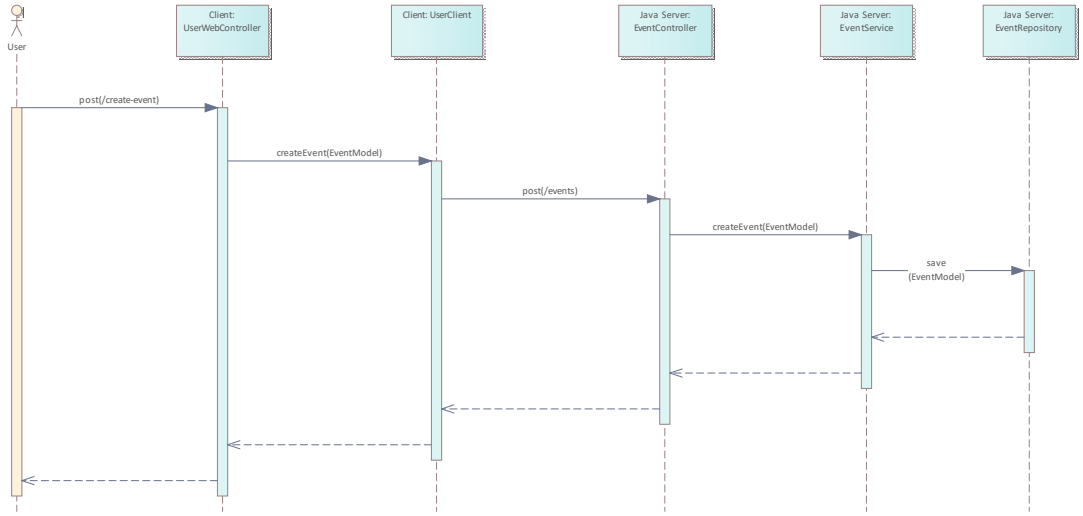

Figure 17 - Event Creation

1.User sends post request to create an event.
2. This method creates client Event Model
3. Post request to send Event Model to server
4. This method created Event Model on server
5. This method saves an Event Model into the database

3.2 List of Events

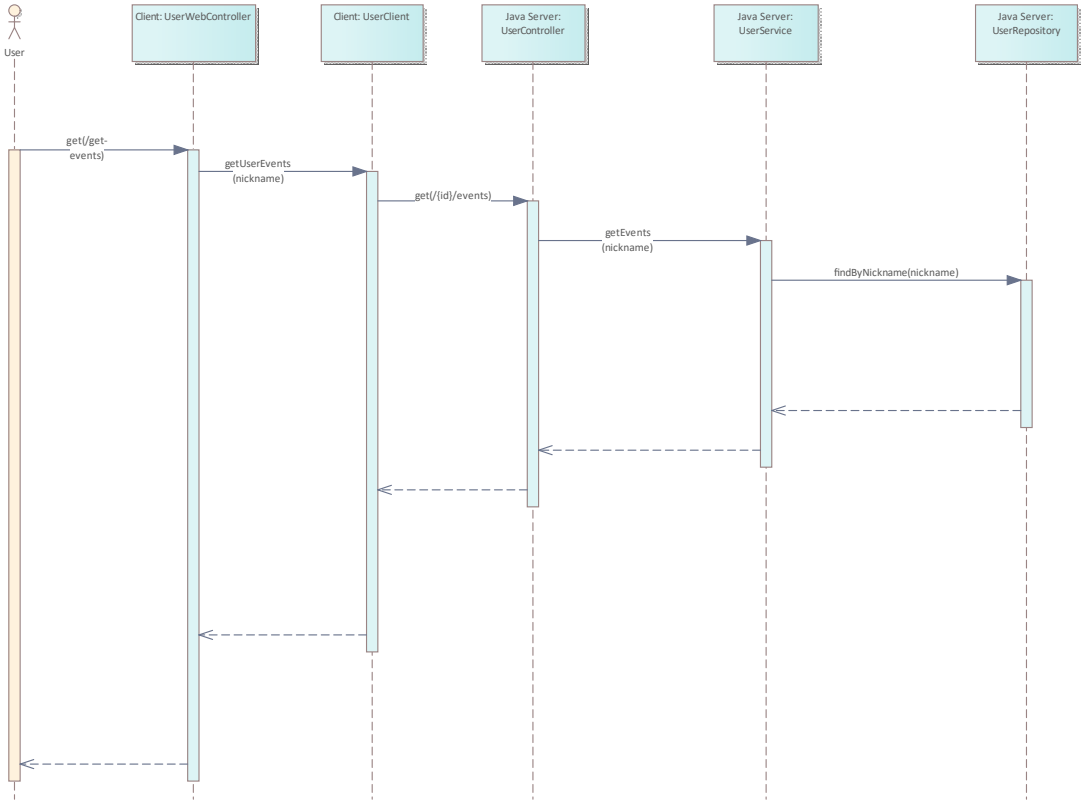

Figure 18 - List of Events

1. Get request to get events for user
2. This method sends a nickname to UserClient class
3. Get request that sends get request to server
4. This method process the nickname in UserController and sends valid nickname to UserService.
5. This method acquires list of Event Models

3.3 User Registration

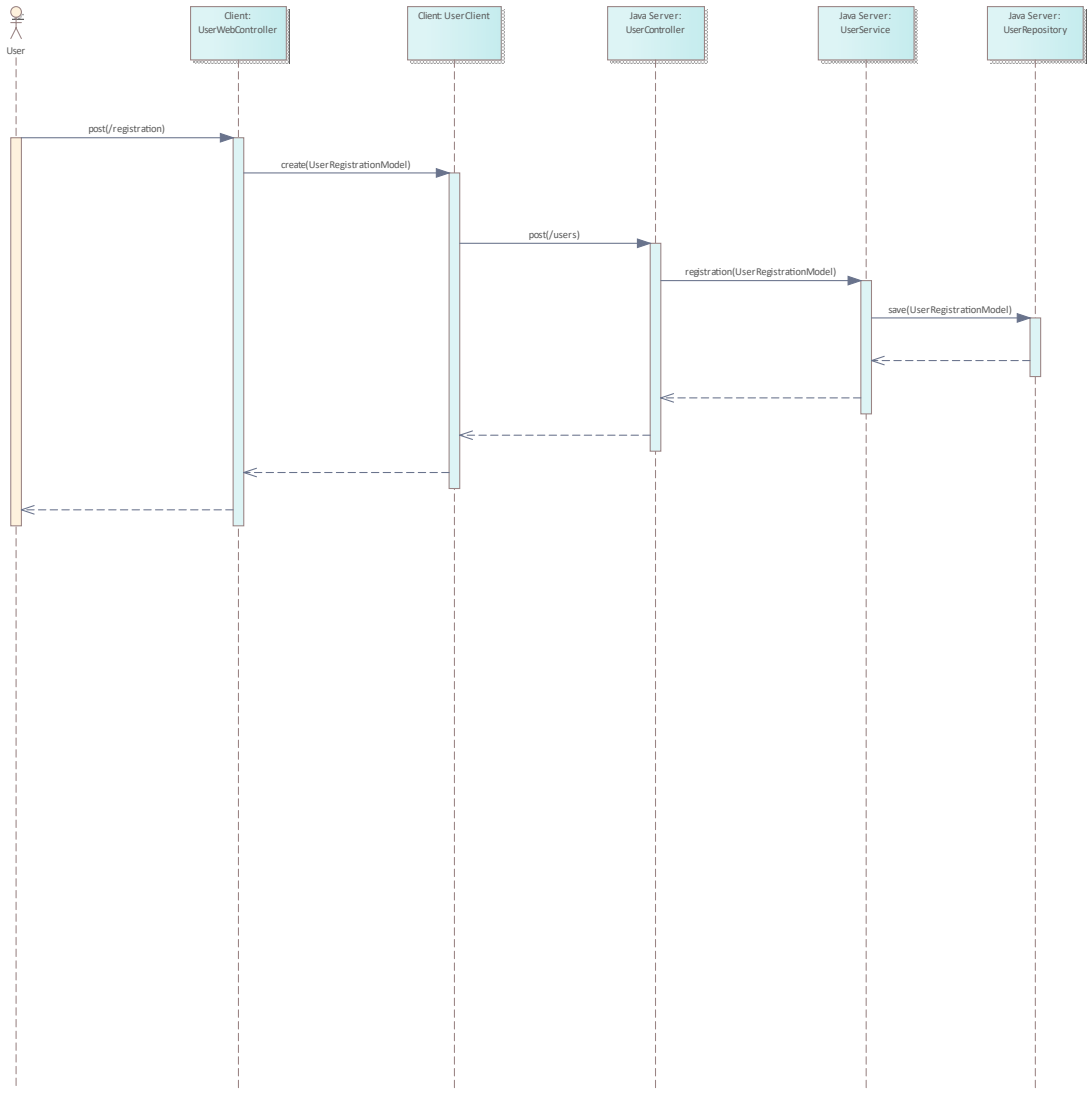

Figure 19 - User Registration

1. Post request for registration
2. This method creates Registration Model in client
3. Post request to create new user.
4. This method accesses business logic which creates registration model on server.
5.This method saves user registration model into the database.