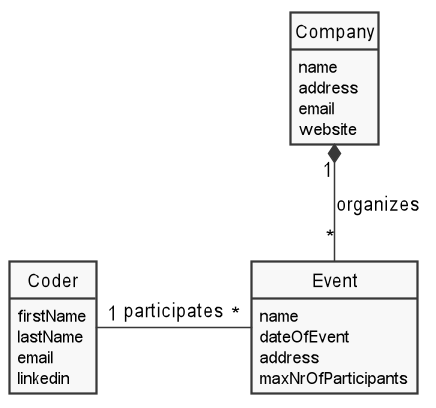
Find a Hackathon

Analysis and Design Document

# Elaboration – Iteration 1.1

# Domain Model



# Architectural Design

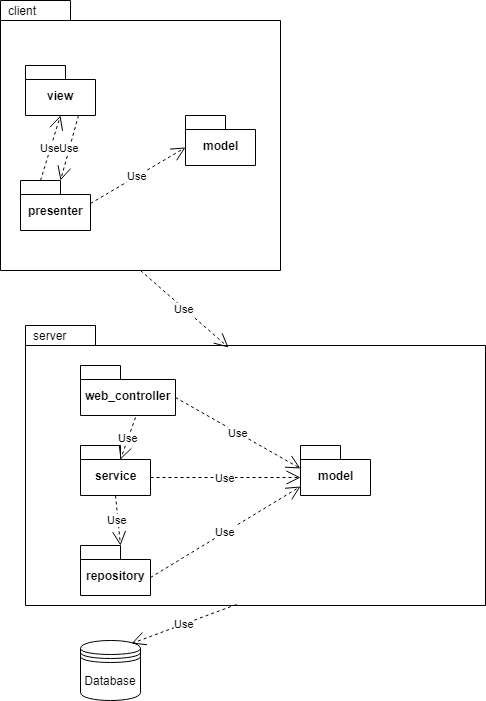
## Conceptual Architecture

This is a client-server web application and the two parts will use different architectural styles, since they provide different types of functionality.

The server will have a layered architecture, with three major layers: web controller, service and repository layer. This allows to change the technology (database, protocol of communication with the client) without affecting the service layer, which implements the business logic.

The client will use a Model-View-Controller architecture, which is best suited for handling interactive user input. The view will only display information, whereas the controller will link the actions received from the view to the model. The model is responsible for communicating with the server. This architecture will keep clean the separation between UI look’n’feel and the functionality provided by the UI, allowing fast change.

## Package Design



## Component and Deployment Diagrams

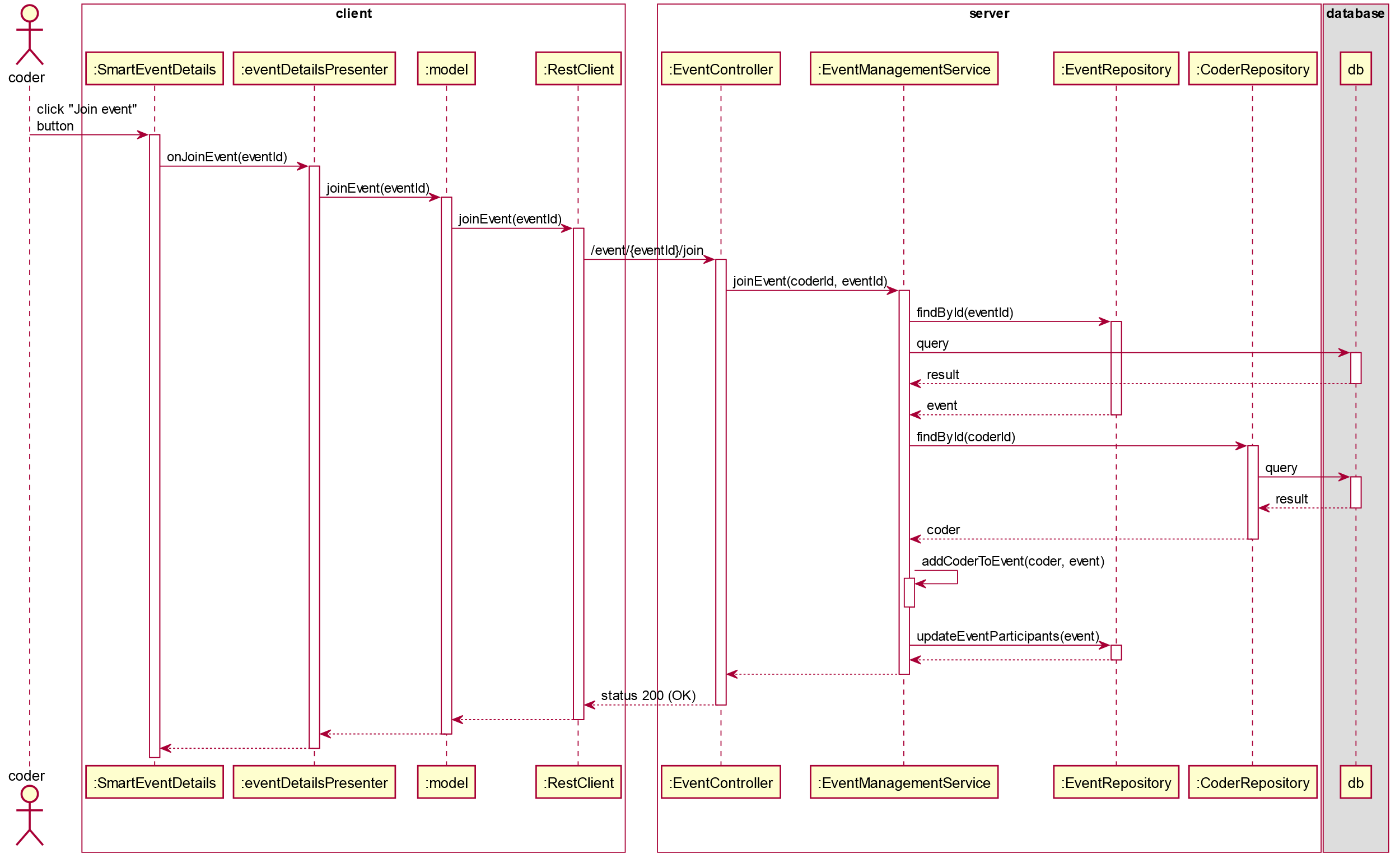
# D:\Google Drive\poze scoala\sd\lab\project-lupvasile\docs\deliverables\component diagram.png

# D:\Google Drive\poze scoala\sd\lab\project-lupvasile\docs\deliverables\deployment diagram.png

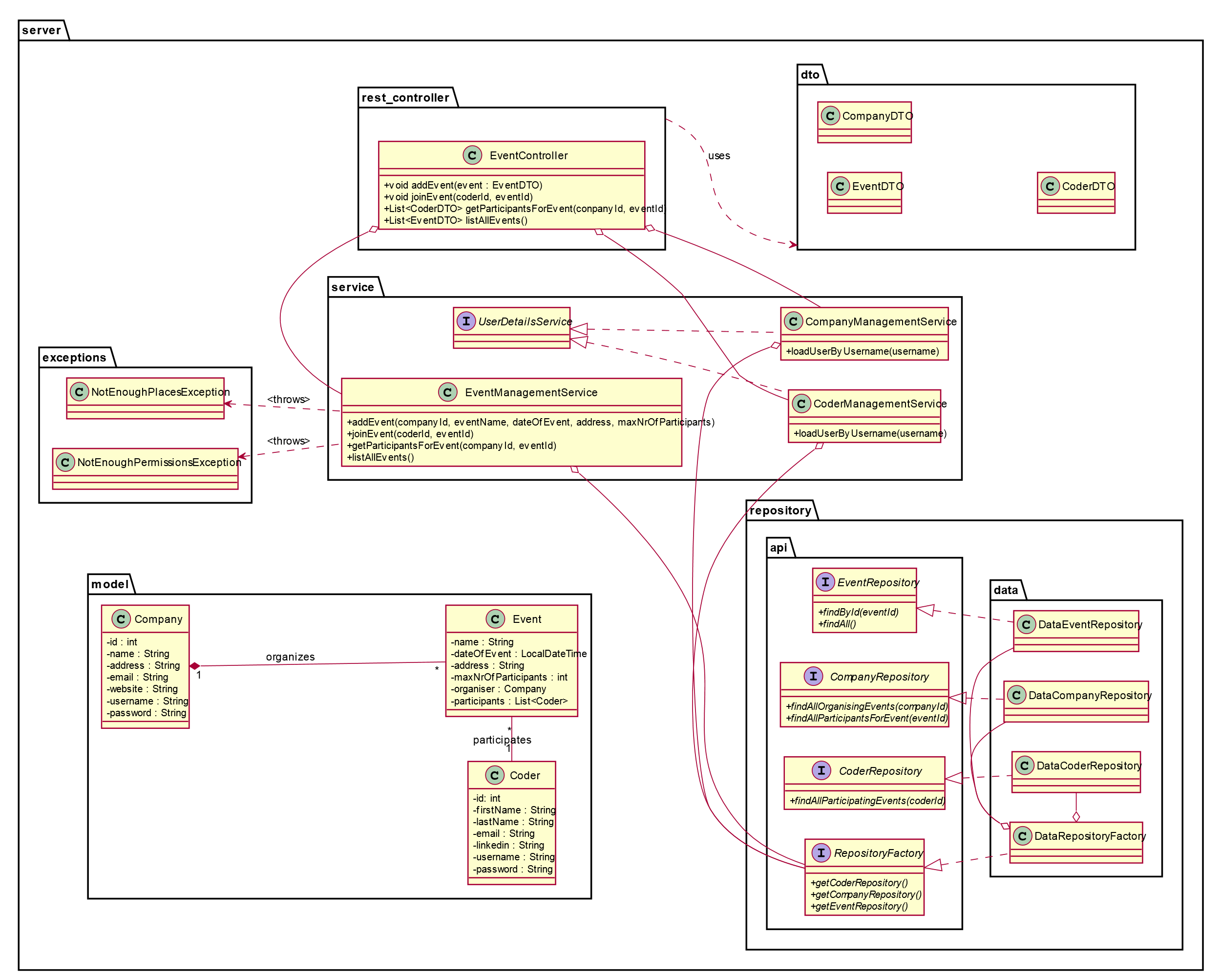
# Elaboration – Iteration 1.2

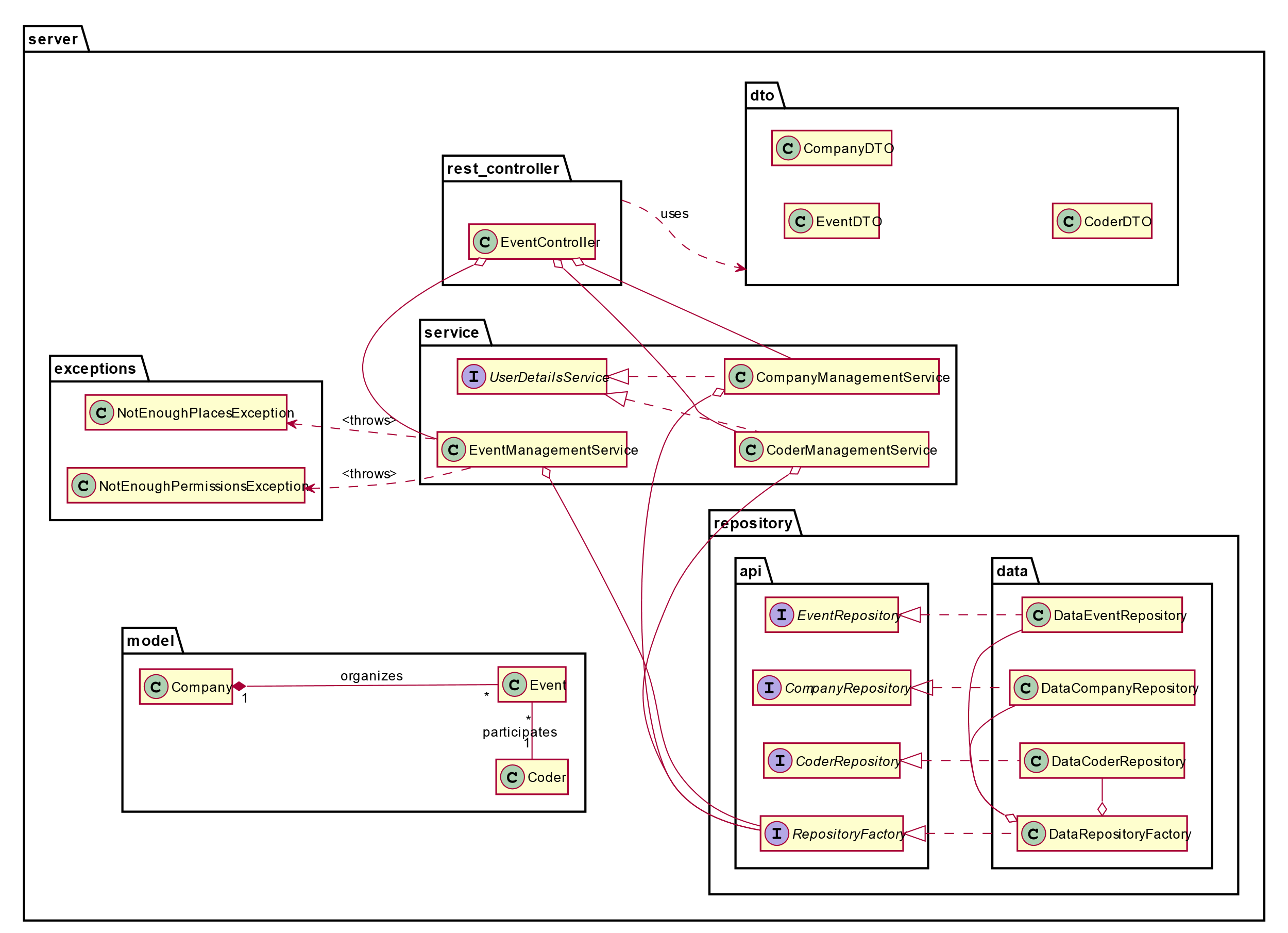
# Design Model

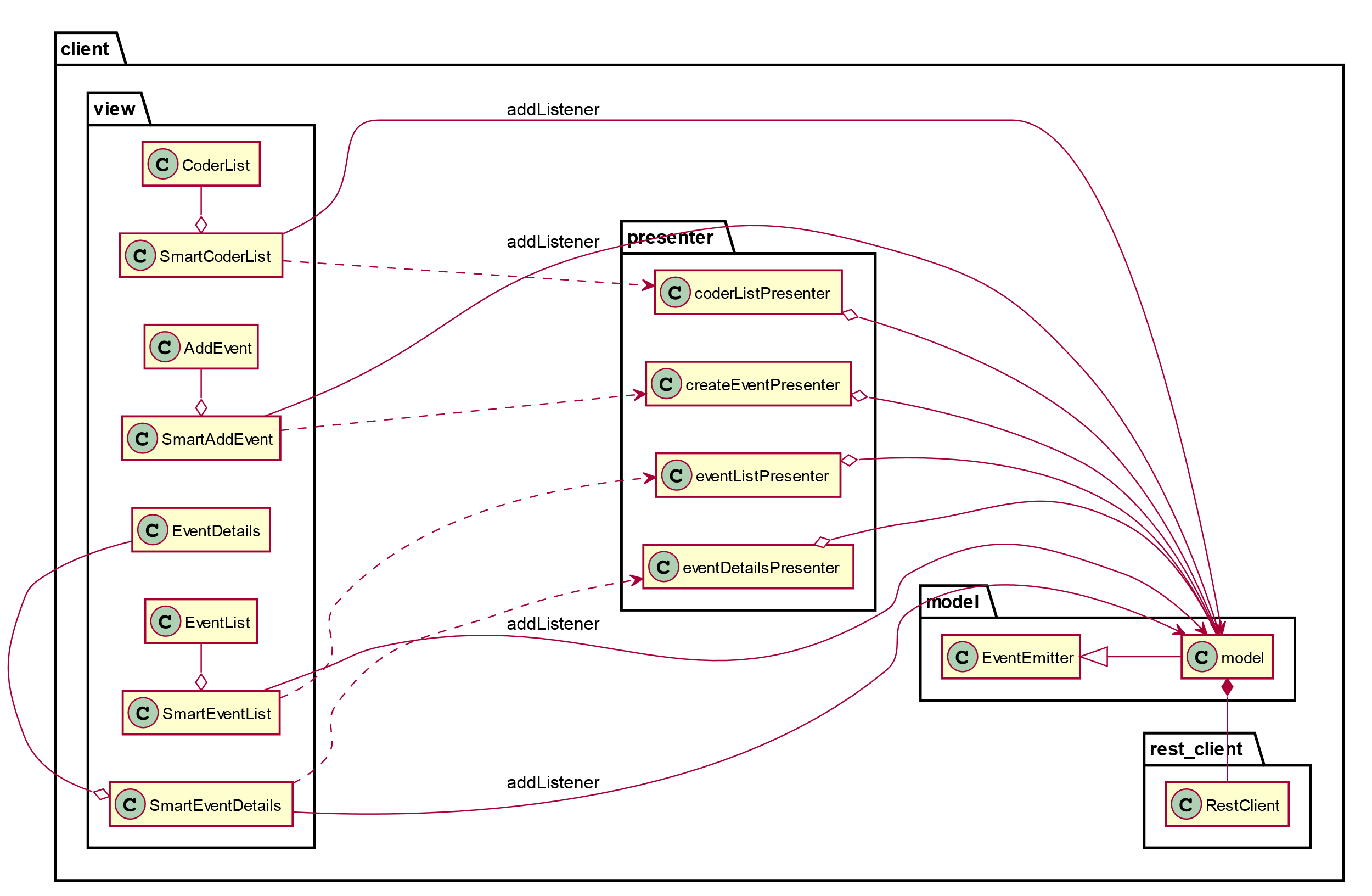
## Dynamic Behavior



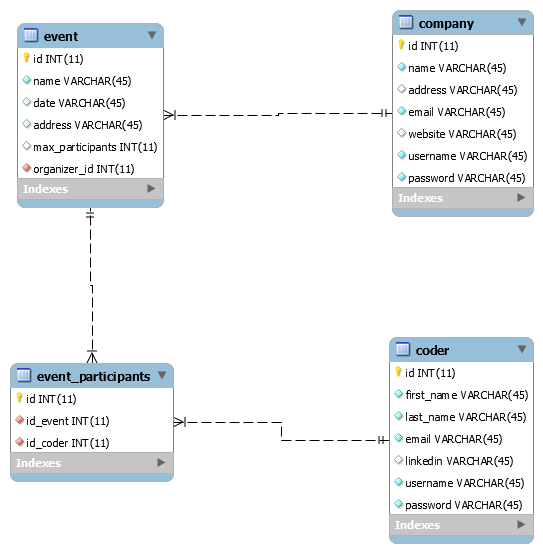
## Class Design







# Data Model



# Test Strategy

There will be three types of tests employed.

1. Unit tests

Each method in classes from the server’s service package need to be tested in unit tests. The coverage for service package needs to be 100%. The methods should be tested in normal flow of control, as well as when throwing exceptions. The database will be an in-memory one, like H2.

Unit tests will also be provided for repository classes.

1. UI tests

The views in the client side will be tested using component tests. The coverage should be 80%.

1. Integration tests

The service layer of the client is tested using the real database and real repositories.

1. End to end tests

The while functionality of app is tested, from a browser. First, logged in a company account, events are created and the list of participants in existing events are checked. Then, logged in a coder account, events are searched, and joined.

The coder will try to join an event with maximum number of participants and he should not be allowed to.

The coder will try to join an event which was due and he should not be allowed to.

The coder will join an event previously added during the testing, and then the tester logins as a company and checks that the coder appears in the event’s list of participants.

# Elaboration – Iteration 2

# Architectural Design Refinement

*[Refine the architectural design: conceptual architecture, package design (consider package design principles), component and deployment diagrams. Motivate the changes that have been made.]*

# Design Model Refinement

## *[Refine the UML class diagram by applying class design principles and GRASP; motivate your choices. Deliver the updated class diagrams.]*

# Construction and Transition

# System Testing

*[Describe how you applied integration testing and present the associated test case scenarios.]*

# Future improvements

*[Present future improvements for the system]*