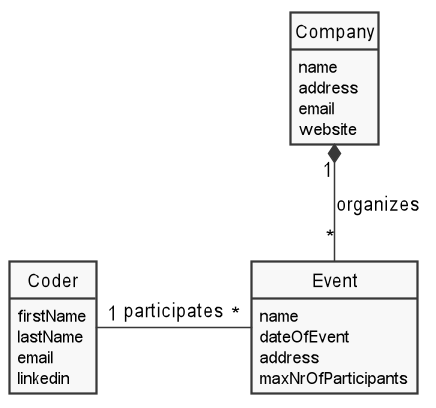
Find a Hackathon

Analysis and Design Document

# Elaboration – Iteration 1.1

# Domain Model



# Architectural Design

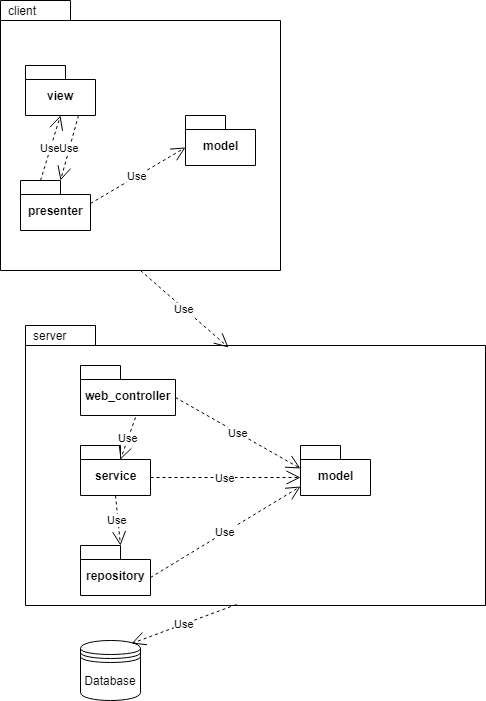
## Conceptual Architecture

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

*[Create the interaction diagrams (1 sequence, 1 communication diagrams) for 2 relevant scenarios]*

## Class Design

*[Create the UML class diagram; apply GoF patterns and motivate your choice]*

# Data Model

*[Create the data model for the system.]*

# Test Strategy

*[Present the used testing methods and the associated test case scenarios.]*

# 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]*