

Full-Stack-Development of the Event-Management-System „PAF-Event“

Simon Detmer

Berliner Hochschule für Technik

<https://github.com/SimonDetmer/PAF-Event>

Abstract

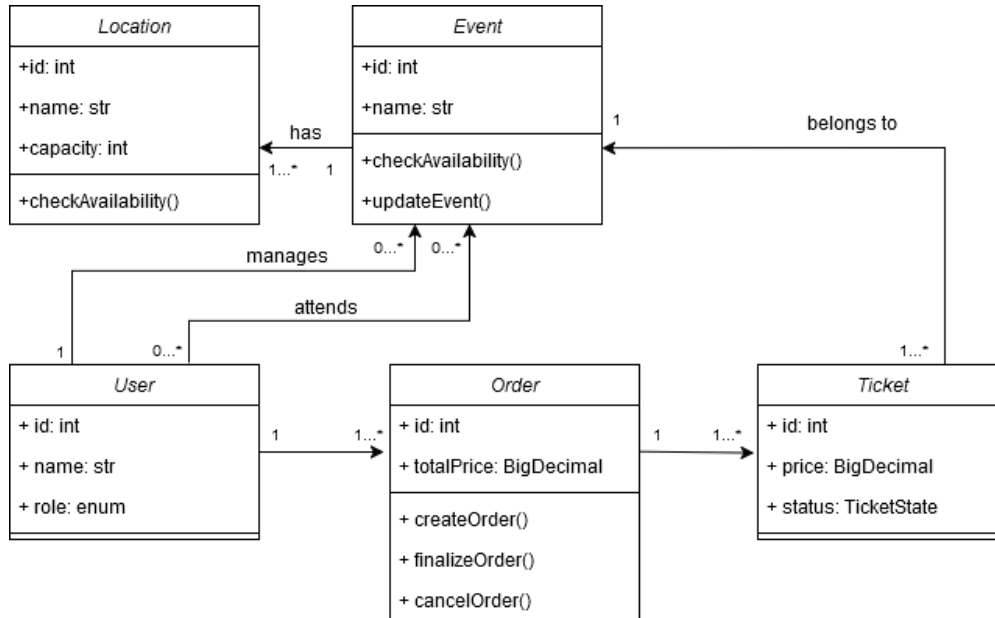
This project involves developing a full-stack web-application for creating and managing events. The aim is to provide an efficient, user-friendly platform that enables Eventmanagers to easily create events and offers customers the option to purchase tickets. The project's primary focus, however, is on the implementation of the ticket purchasing process and related concurrency challenges. Particular attention is given to preventing collisions and handling errors as they arise.

Table of contents

1	Introduction	2
2	Domain Model	2
3	User Stories	2
4	Backend-Components	3
5	Pattern	3
6	Special Topic - Backend: Concurrency	3
7	Testing	3
7.1	Backend	3
7.2	Frontend e2e with Cypress	3
8	Frontend Logic	3
9	Special Topic - Frontend: eg. Web Sockets	3

1 Introduction

2 Domain Model



3 User Stories

1. As an event eventmanager, I want to create an event with details like name and location so that customers can book tickets for it.
2. As an event eventmanager, I want to assign locations to events so that each event has a venue with a capacity limit.
3. As an event eventmanager, I want to edit event details after creation to reflect any changes.
4. As an eventmanager, I want to view ticket sales reports so that I can track event revenue.
5. As a customer, I want to see a list of upcoming events so that I can decide which one to attend.
6. As a customer, I want to view my past orders so that I can keep track of the events I've attended.
7. As a customer, I want to add tickets to my order and purchase them so that I can attend the event.
8. As a customer, I want to receive meaningful error messages when something goes wrong, such as when a ticket is unavailable, so that I understand the issue.

4 Backend-Components

5 Pattern

6 Special Topic - Backend: Concurrency

7 Testing

7.1 Backend

7.2 Frontend e2e with Cypress

8 Frontend Logic

½ - 1 Seite: Services, Model-Classes, Utils, Handler etc.

2 Seiten MockUps abgeleitet aus User Stories

9 Special Topic - Frontend: eg. Web Sockets