Sportify
Software Architecture Document

Version 2.0

Revision History

Date	Version	Description	Author
<23/10/22>	<1.0>	First draft	Firangiz Ganbarli
<27/10/22>	<2.0>	Updated the design class diagram and descriptions	Firangiz Ganbarli

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. Is it intended to capture and convey the significant architectural decisions which have been made on the system.

1.2 Scope

This Software Architecture Document provides an architectural overview of Sportify. Sportify will allow users to register and create sport events and will also allow users to search for events that are around them and show the results of that query.

1.3 References

1. Sportify – Use Case Specification
2. Sportify – Supplementary Specification
3. Sportify – Use Case Realization

2. Architectural Representation

This document presents the architectural as a series of views: use case view, process view, deployment view, and implementation view. These views are presented as Rational Rose Models and use the Unified Modeling Language (UML).

3. Architectural Goals and Constraints

Sportify is to be developed as a stand-alone web app that will be accessible by any major Internet Browsers. It consists of four key components: a User Client Module, a Server Module, a Database, and an Administrator Client Module. All components must be able to execute on a personal computer. The User Client and Administrator Client modules must communicate with the server over a TCP/IP connection. The Server and Database components should be located on the same host.

4. Use-Case View

The Use Case View is crucial input to the selection of the set of scenarios and use cases that are the focus of an iteration. It describes the set of scenarios and use cases that represent some significant, central functionality. It also describes the set of scenarios and use cases that have many architectural elements.

For more information, refer to Use-Case Specification Requirements document.

4.1 Use-Case Realizations

Refer to Use Case Realization document.

5. Logical View

5.1 Overview

This subsection describes the overall decomposition of the design model in terms of its package hierarchy and layers.

5.2 Architecturally Significant Design Packages

5.2.1 Design Model: Packages Diagram

The design model represents the structure and organizations of Sportify. Packages and classes are presented with a brief description.

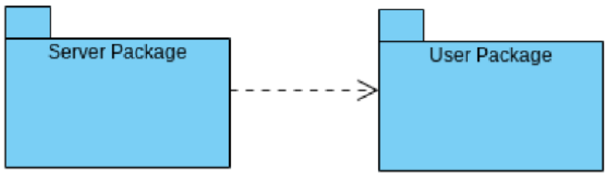

Figure 1: Design Model Package

5.2.2 Design Model: Packages Description

Server Package	
Description	This is the main system package where all client queries are managed
Corresponding Classes	Observable, Observer, SportifyServer, DeveloperObserver, AdministratorObserver
Relations	Main package, Dependent of: User
Sub-Packages	Users

User Package	
Description	All info regarding users are handled in this package
Corresponding Classes	User, Event, EventDatabase, SignedUpEvent, SearchResults
Relations	Sub package of Server
Sub-Packages	None

5.2.3 Design Model: Design Class Diagram

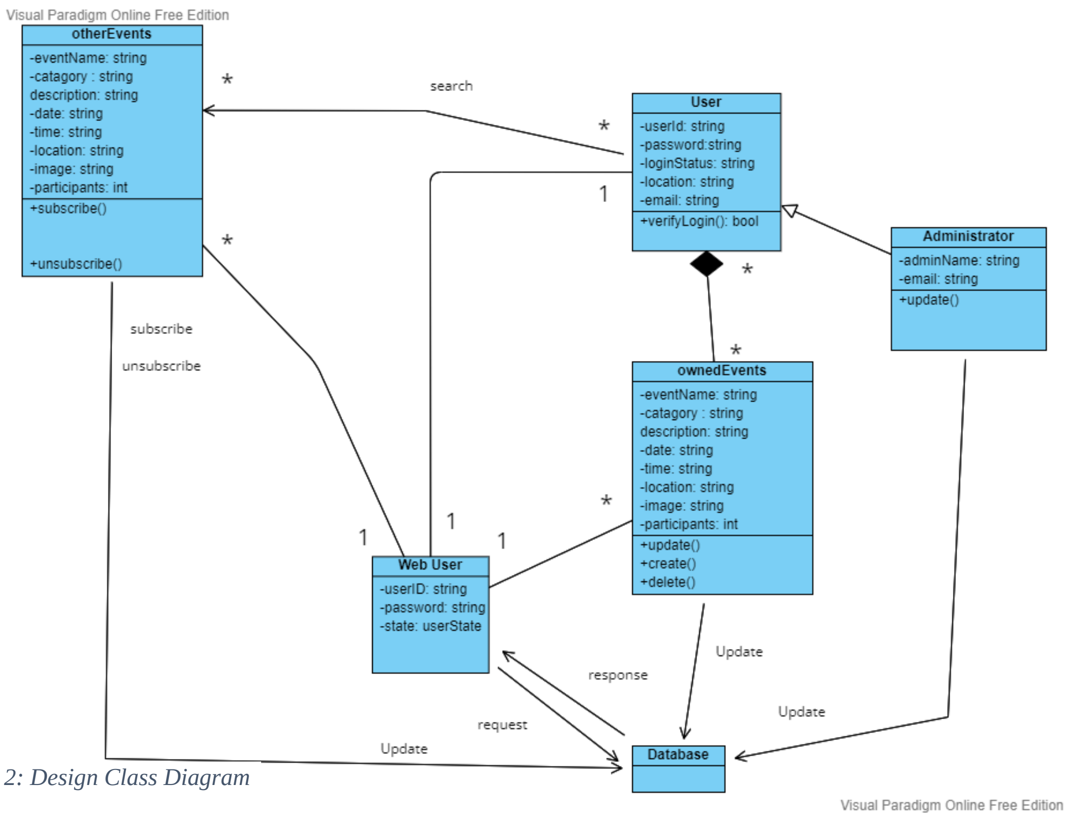

Figure 2: Design Class Diagram

5.2.4 Design Classes Description

Property	Description
Name	User
Description	Represents the User entity
Responsibilities	Maintains and updates all the private User information
Relations	Generalization with Administrator, Composition with ownedEvents of multiplicity *, one way association with otherEvents of multiplicity * and association with Web User.
Attributes	userID: the username of the user password: password the user sets up loginStatus: whether login was successful or not location: the city user is based in email: user's email
Methods	verifyLogin(): verifies the login information and authenticates the user

Property	Description
Name	Administrator
Description	It's a class that represents the SportifyServer, deals with the user and database.
Responsibilities	Communicates between the user and the database the server uses.
Relations	Generalization with User, One-way Association with Database
Attributes	adminName: the name of the administrator email: email of the admin acocunt
Methods	update(): updates the user information accordingly to the database

Property	Description
Name	ownedEvents
Description	It's a class that represents events that host has created.
Responsibilities	Allows the event host to manage the event information page.
Relations	Composition with User of multiplicity *, One-way association with database, and an association with a web user of 1.
Attributes	eventName: title of the event category: the sport category of the event description: short description of the event date: day of the event time: time of the event location: the address the event is located at image: optional image to put as a background image participants: list of users who signed up for the event
Methods	update(): lets the host edit event information create(): lets users create a new event delete(): lets hosts delete their event

Property	Description
Name	Database
Description	Class that represents where the event and user data are located at
Responsibilities	Contains all the event and user information in a secure database
Relations	Association with ownedEvents, Administrator and WebUser
Attributes	none
Methods	none

Property	Description
Name	Web User
Description	Represents an active user
Responsibilities	Allows a logged in user access various features of the app
Relations	Association with otherEvents with multiplicity of *, association with database, association with User with multiplicity of 1, and an association with ownedEvents with multiplicity of 1.
Attributes	userID: the username of the user password: password user has set up for login state: the activity state of the user
Methods	none

Property	Description
Name	otherEvents
Description	Class that represents events that users have access to see and sign up for
Responsibilities	Contains all the event info and lets users sign up or unregister for an event.
Relations	Association with User with multiplicity of *, and association with 1 web user, and an association with database
Attributes	eventName: title of the event category: the sport category of the event description: short description of the event date: day of the event time: time of the event location: the address the event is located at image: optional image to put as a background image participants: list of users who signed up for the event
Methods	subscribe(): lets users sign up for an event unsubscribe (): lets users unregister for an event they have already signed up for

6. Interface Description

Refer to the files within Sportify -> Prototype folder.

7. Size and Performance

The selected architecture supports the sizing and timing requirements. The website components have been designed to ensure that it takes minimal time to load and does not slow down the browser significantly.

8. Quality

The software architecture supports the quality requirements, as stipulated in the Software Requirements Specifications and Supplementary Specification.