|  |  |
| --- | --- |
|  |  |

Ingegneria del Software e Progettazione Web

Progetto A.A. 2022/2023

SPORTIFY

Chiara D’Ambrogio

Matteo La Gioia

**Indice**

[1. Software Requirement Specification 3](#_Toc122535472)

[1.1. Introduction 3](#_Toc122535473)

[1.1.1. The aim of the document 3](#_Toc122535474)

[1.1.2. Overview of the defined system 3](#_Toc122535475)

[1.1.3. Operational settings 3](#_Toc122535476)

[1.1.4. Related systems 4](#_Toc122535477)

[1.2. User Stories 4](#_Toc122535478)

[1.3. Functional Requirements 5](#_Toc122535479)

[1.4. Use cases: Overview Diagram 6](#_Toc122535480)

[2. Storyboards 7](#_Toc122535481)

[2.1. Format: HTML 7](#_Toc122535482)

[2.2. Screens 7](#_Toc122535483)

[3. Design 8](#_Toc122535484)

[3.1. Class diagram 8](#_Toc122535485)

[3.2. Design patterns 8](#_Toc122535486)

[3.3. Activity diagram 8](#_Toc122535487)

[3.4. Sequence diagram 8](#_Toc122535488)

[3.5. State diagram 8](#_Toc122535489)

[4. Testing 9](#_Toc122535490)

[4.1. Test Cases 9](#_Toc122535491)

[4.2. Test Selenium GUI 9](#_Toc122535492)

[4.3. Test Selenium API 9](#_Toc122535493)

[5. Code 10](#_Toc122535494)

[5.1. ~4K LOC 10](#_Toc122535495)

[5.2. Similar functionality implemented with GUIs. 10](#_Toc122535496)

[5.3. Exceptions 10](#_Toc122535497)

[5.4. Svn (or Git) + SonarCloud 10](#_Toc122535498)

[5.5. DAO 10](#_Toc122535499)

[6. Analytics 11](#_Toc122535500)

[7. Video 12](#_Toc122535501)

# Software Requirement Specification

## Introduction

### The purpose of this document is to provide information.

The project aims to encourage people to practice a sport, helping them choose the best sport for them and the gym where to practice it.

### Overview of the defined system

Sportify is used by users to search for courses and gyms and for the latter also leave reviews on their service, while by gyms to be able to add courses and schedules of the latter to their Sportify profile, now let us describe the guaranteed features:

* User:
* Search for sports
* Fill out the test to understand the sport that best suits you.
* Find your nearest gym
* Writing reviews of popular gyms
* Gyms:
  + Add courses and timetables for these last
  + Read reviews of your gym
  + Delete possible unconstructive reviews.

### Operational settings

There are many operating settings for Sportify; the program will be used as a mobile application or computer program by users looking for courses or gyms, by users who want to understand the most suitable sport for them, and by gyms that want to advertise themselves to users.

For the development of the software, we used the following:

* Figma.com to get the HTML code of the storyboards.
* Star UML for the realization of diagrams related to the design of the system.
* Scene Builder to create FXML files to create the graphical interface (MVC views).
* JavaFX to manage and extend FXML files (MVC graphics controller).
* IntelliJ and especially Java as an IDE and programming environment. To use the software, you must extract the Sportify folder from the zip file and open it as a project in IntelliJ. In the folder there is already the ‘.idea’ file to configure how to run the application. If this is not possible, the class responsible for starting the application is MainAppLauncher.java.

### Related Systems

(at least 2), Pros and Cons.

Making a careful analysis of the competitors, although there are few similar services, we have identified the two in particular: *nome1*, *nome2*.

• *Nome1*: *Cosa offre, in cosa è meglio Sportify*.

• *Nome2*: *Cosa offre, in cosa è meglio Sportify*.

## User storeys

(3 per member)

**• As a new user, I want to discover which sport suits me best and my needs so that I can start practicing it.**

**• As a user, I want to review my experience in a gym so that it could be useful to other users.**

**• As a gym owner, I want to know which are the ten most popular sports in Italy, so that I can add them to my gym offer.**

**• As a new user, I want to know which gyms have a selected sport in a variety of choices so that I do not waste time moving.**

**• As a user, I want to know the main information about a selected sport, so that I can decide whether to practice it.**

**• As a gym, I want to sponsor myself to gain greater visibility.**

• As a gym owner, I want to read the reviews written about my gym, so that I can modify and improve my gym offer.

• As a user, I want to keep track of what level I reached in the sports I play, so that I can continuously improve.

## Functional requirements

(3 per member)

• **The system, based on specific questions, must decide the best sport for the user’s needs.**

**• The system must provide the ability to read user reviews.**

**• The system shall provide all gyms in a range of choice from a chosen location in which they can practice the selected sport.**

**• The system creates a list of the ten most practiced sports in Italy based on the number of times a sport has been chosen.**

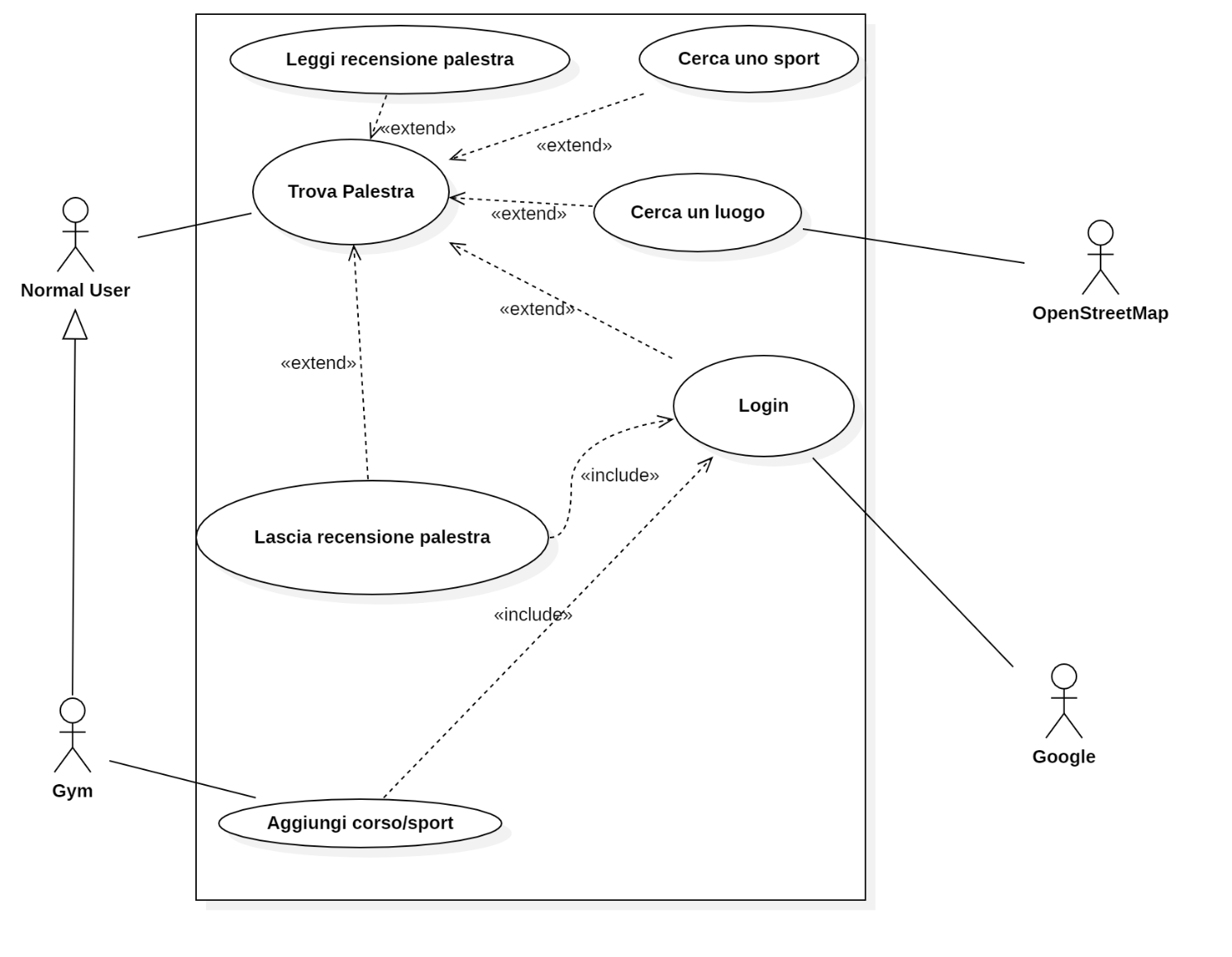
**• The system must provide the main information about a selected sport.**

**• The system sends an alert to the user who is searching for a particular sport from gyms that offer that sport and that are in a range of choice from a selected location.**

• The system must keep track of the level at which the user reaches in each sport.

• The system must provide a review of the user experience in the gyms by adding a comment in the appropriate box on the gym profile pages.

## Use cases: Overview Diagram



# Storyboards

## Format: HTML

## Screens

Quantity: two screens per member, covering all the functionalities described in SRS, developed using Draw.io or similar

# Design

## Class diagram

* 1 VOPC per member. (analysis)
* One design-level diagram per member (e.g., that includes patterns or specific solutions that improve the engineering level of the system)

## Design Patterns

One different pattern per member. Possibly, try applying the pattern within the context of the project.

## Activity diagram

One per member.

## Sequence diagram

One per member.

## State diagram

One per member.

# Testing

## Test Cases

Develop at least three test cases per person. In each test (class) file, report (via Java comments) the name of the person in charge.

## Selenium GUI Test

1 Selenium test using the GUI per member.

## Test Selenium API

1 Selenium test through API per member.

# Code

## ~4K LOC

## Similar functionality implemented with GUIs.

## Exceptions

at least two per member (do not just catch and back-propagate the exceptions, but effectively manage them. Possibly define your own error logic by means of exceptions).

## Svn (or Git) + SonarCloud

Be able to show that Svn (or Git) + SonarCloud is correctly installed on one of your computers and it can analyse your project for rule violations. No rule must be violated (no smells, no vulnerabilities, no bugs. This will be checked during the exam.

## DAO

One DAO shall be provided in two versions, DMBS and file system.

# Analytics

Provide a process control chart as explained in the slides.

# Video

A 1- to 2-minute recorded video of the developed system performing the expected functionalities. \*.mpeg