Tietokantasovellus

Gabriel Lindström
August 2013

Contents

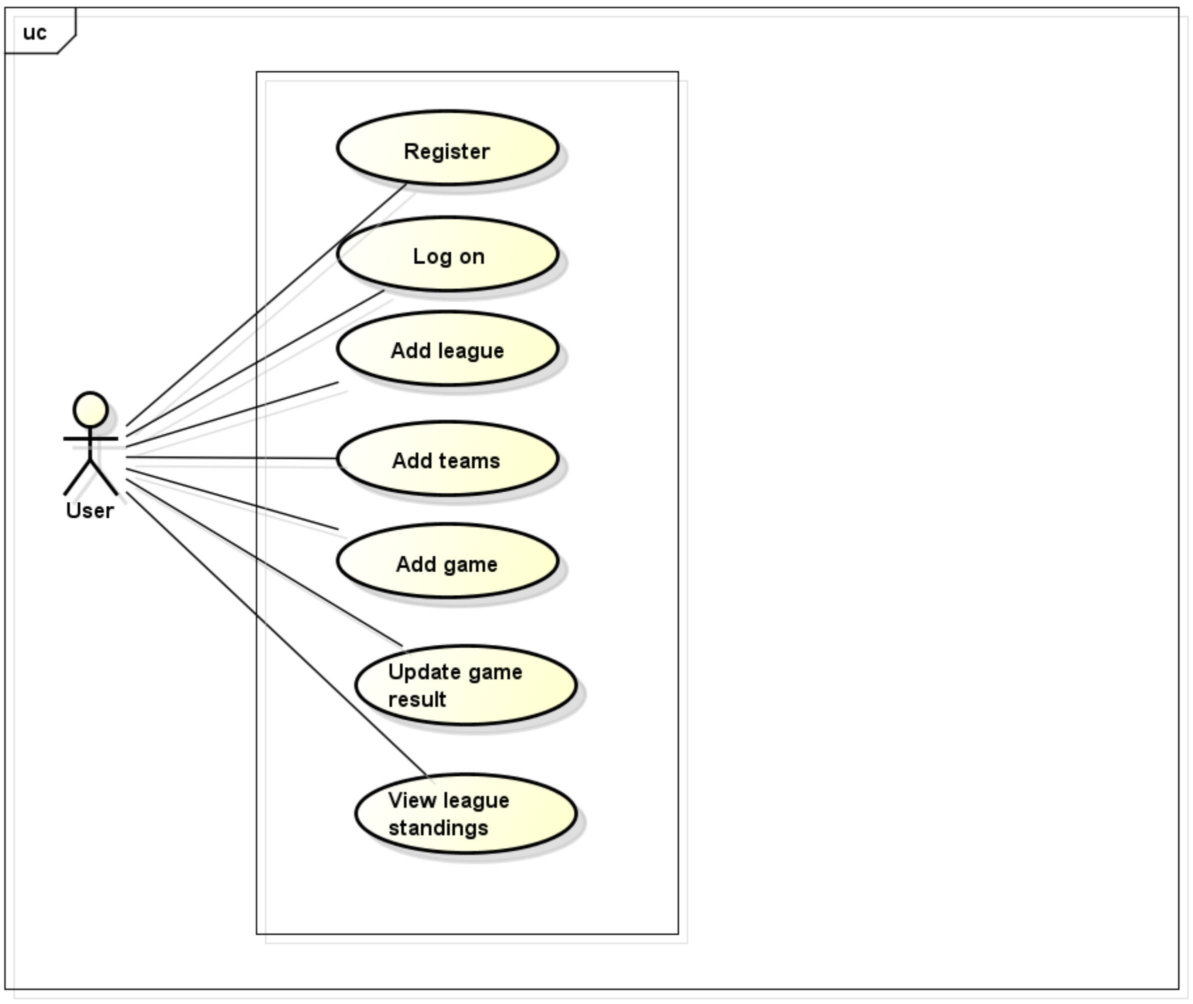
1 Introduction

The aim is to create a web application for storing and viewing football results. The user will be able to add new football leagues and add match results to existing leagues. The user can view the results of individual matches and view the league standings.

The application will be implemented in Java and powered by Apache Tomcat. PostgreSQL will be used as the database system. For the best user experience, Google Chrome is recommended as a browser.

2 System Overview

2.1 Use Case Diagram



2.2 User Groups

<u>User</u>
A user is a registered user of the application.

3 Use Cases

Register:

The user registers for the service by choosing a user name and password.

Log in:

The user logs in to the system using the chosen user name and password.

A prerequisite for all use cases below is that the user has registered logged on to the system.

Add league:

The user submits the name of the league.

Add teams:

The user specifies the name of one or more teams.

Add season:

The user specifies the name of the season.

Add game:

The user specifies the league, season and date of the match, picks the teams playing and optionally enters the result.

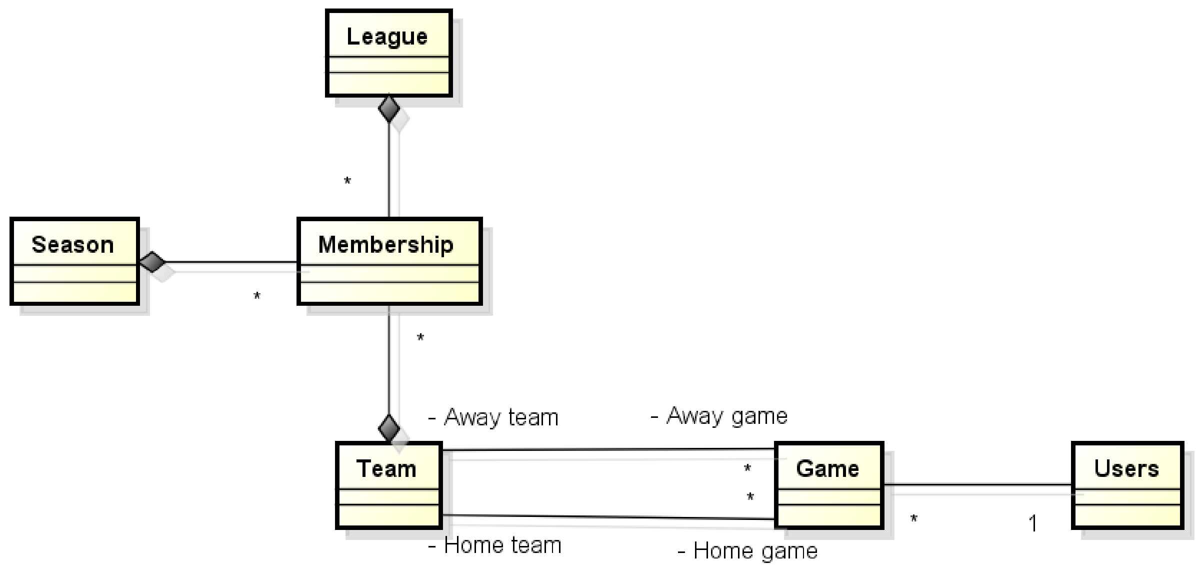
Update game result:

The user adds or updates the scores of the teams.

View league standings:

The user can review the current standings of a specific league and season. The standings show how many games each team has played, aggregated scored and conceded goals and points.

4 Entities



Entity: League

Attribute	Domain	Description
name	varchar, max 30 characters	the name of the league
id	serial	primary key

Every membership is associated with one league and a league can be associated with any number of memberships.

Entity: Membership

A membership defines a team's participation in a league for a specific season. Every membership is associated with exactly one league, team and season whereas a league, a team or a season can be associated with any number of memberships.

Entity: Season

Attribute	Domain	Description
id	serial	primary key
name	varchar, max 9 characters	the name of the season

Every membership is associated with one season and a season can be associated with any number of memberships.

Entity: Team

Attribute	Domain	Description
name	varchar, max 30 characters	the team name
id	serial	primary key

Every membership is associated with one team and a team can be associated with any number of memberships. In a match one time plays as home team and another team as away team. Teams can play many matches.

Entity: Game

Attribute	Domain	Description
id	serial	primary key
home_score	integer	the number of goals that the home team scored
away_score	integer	the number of goals that the away team scored
game_date	date	the date of the match in the format 31-01-2013
time_edited	timestamp	the time when the match record was last edited

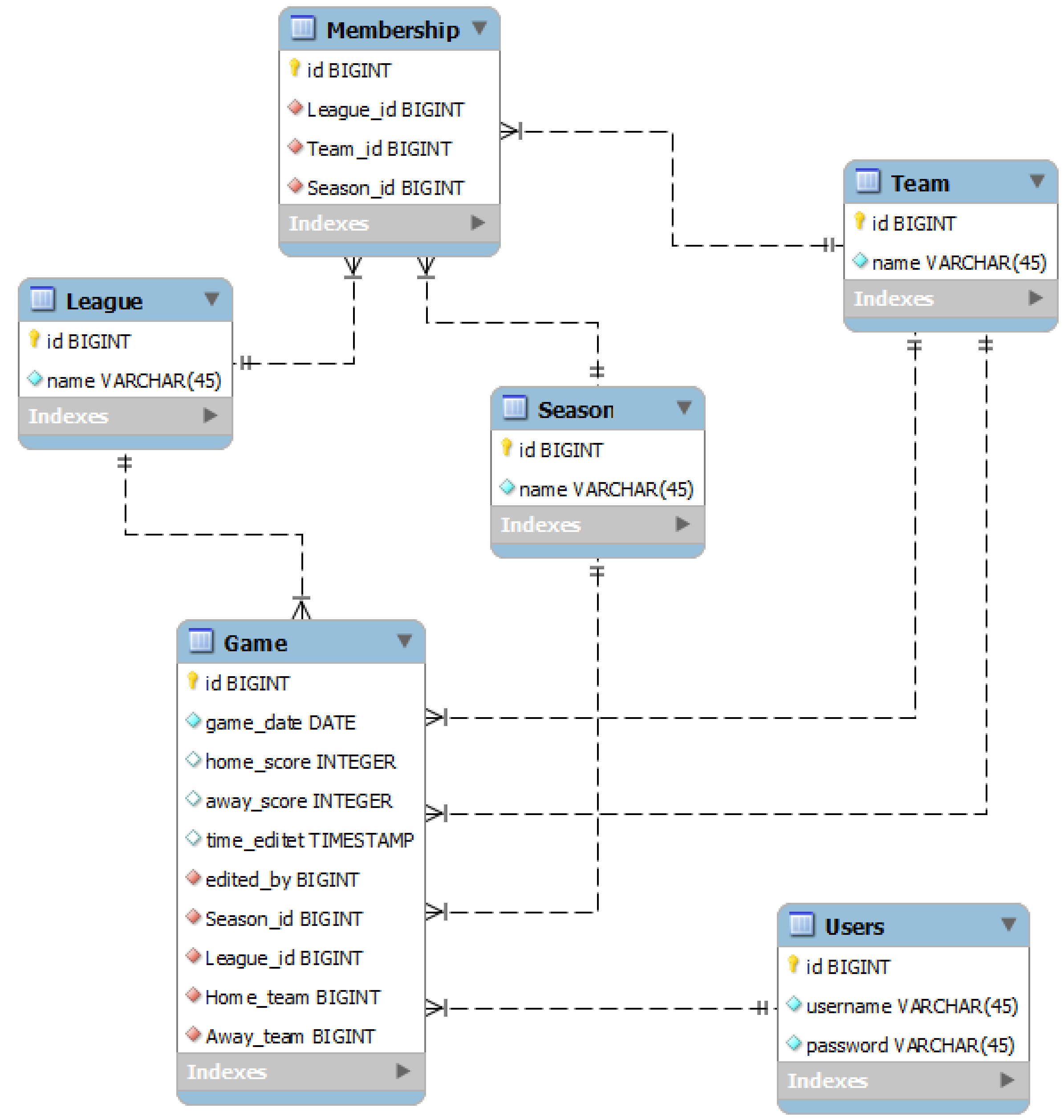
In a match one time plays as home team and another team as away team. Teams can play many matches. Each match record is associated with the user who last edited the record. A user can edit many match records.

Entity: Users

Attribute	Domain	Description
id	serial	primary key
Username	varchar	
Password	varchar	

A user must register with a username and password in order to use the service. Matches are associated with the last user who edited the match record.

5 Entity Relationship Diagram



The attribute types in the diagram differ slightly from the ones used in the database as the program user for creating the diagram did not support all postgresql data types. More precisely, in the diagram bigint is used instead of serial and varchar(45) replaces varchar. The create statements can be found in the repository's sql folder.

6 System Structure

I have implemented the model-view-controller design pattern for this project. Furthermore, the database connection has been handled through the data access object design pattern. The part pertaining to the view consists of JSP files, which are located in the web folder. The controllers, i.e. servlets are in the servlets package. The models package contains classes, which correspond to particular tables and queries. The dao package contains classes for handling the database operations for each model. The dbconnection package contains a class, which provides a connection to a specified database. External libraries are placed in the lib folder.

7 System Components

addmembership.jsp

Memberships are added using drop down lists.

addseason.jsp

A new season can be created by submitting a name in an input field.

addteam.jsp

A new team can be created by submitting a name in an input field.

games.jsp

Shows a table of all games and offers the possibility to update, delete or add a new game.

leagues.jsp

Shows a list of all leagues and offers the possibility to delete or add a new league.

login.jsp

The user can log in by entering the username and password in a form.

memberships.jsp

Shows a list of all memberships and offers the possibility to delete or add a new membership.

register.jsp

A user can register by filling in the desired username and password.

seasons.jsp

Shows a list of all seasons and offers the possibility to delete or add a new season.

standings.jsp

style.css

Style sheet used on the login page.

teams.jsp

> Shows a list of all teams and offers the possibility to delete or add a new team.

standings.jsp

> Shows the standings table for the selected league and season.

welcome.jsp

> The first page the user sees after logging in. Contains a menu of all functions.

viewstandings.jsp

> The user can select the league and season for which to display the standings table.

addgame.jsp

> Page for adding a new game. The user chooses the league and season In which the game takes place.

addgame2.jsp

> On this page the user selects the date of the game, the home team, the away team and optionally enters the score of the game. The teams available in the drop down list are depending the previously selected league and season.

addleague.jsp

> A new team can be created by submitting a name in an input field.

updategame.jsp

> The score can be corrected or inserted if it's missing.

src\java\fi\cs\helsinki\glindstr\soccerdb\servlets\RegisterServlet.java

> This servlet handles the register functionality.

src\java\fi\cs\helsinki\glindstr\soccerdb\servlets\SeasonServlet.java

> This servlet handles the management of seasons.

src\java\fi\cs\helsinki\glindstr\soccerdb\servlets\StandingsServlet.java

> This servlet handles the standings view.

src\java\fi\cs\helsinki\glindstr\soccerdb\servlets\TeamServlet.java

> This servlet handles the management of teams.

src\java\fi\cs\helsinki\glindstr\soccerdb\servlets\GameServlet.java

> This servlet handles the managing of games.

src\java\fi\cs\helsinki\glindstr\soccerdb\servlets\LeagueServlet.java

This servlet handles the management of leagues.

src\java\fi\cs\helsinki\glindstr\soccerdb\servlets\LoginServlet.java

This servlet handles the logging in and out of the system.

src\java\fi\cs\helsinki\glindstr\soccerdb\servlets\MembershipServlet.java

This servlet handles the managing of memberships.

src\java\fi\cs\helsinki\glindstr\soccerdb\models\Standing.java

This class represents a row of the standings query.

src\java\fi\cs\helsinki\glindstr\soccerdb\models\Team.java

This class represents a record of the team table.

src\java\fi\cs\helsinki\glindstr\soccerdb\models\User.java

This class represents a record of the users table.

src\java\fi\cs\helsinki\glindstr\soccerdb\models\Game.java

This class represents a record of the game table.

src\java\fi\cs\helsinki\glindstr\soccerdb\models\League.java

This class represents records of the league table.

src\java\fi\cs\helsinki\glindstr\soccerdb\models\Membership.java

This class represents a record of the membership table. A membership defines which teams play in a league a certain season.

src\java\fi\cs\helsinki\glindstr\soccerdb\models\Season.java

This class represents a record of the season table.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\BaseDao.java

An interface for database operations. The BaseDao interface provides a method for inserting records to the database and a method for deleting records from the database.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\StandingsDao.java

An interface for retrieving the standings from the database. Provides a method for getting a list with all standings.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\StandingsDaoImpl.java

This class retrieves the standings from the database.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\TeamDao.java

An interface for database operations on the team table. The TeamDao interface provides a method for gaining a list of all teams.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\TeamDaoImpl.java

This class provides database access for the team table.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\UserDao.java

An interface for database operations on the users table. The UserDao interface provides a method for validating a user.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\UserDaoImpl.java

This class provides database access for the user registration function.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\GameDao.java

An interface for database operations on the game table. The GameDao

interface provides a method for gaining a list of all games.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\GameDaoImpl.java

This class provides database access for the game table.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\LeagueDao.java

An interface for database operations on the league table. The LeagueDao interface provides a method for gaining a list of all leagues

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\LeagueDaoImpl.java

This class provides database access for the league table.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\MembershipDao.java

An interface for database operations on the membership table. The MemebershipDao interface provides a method for gaining a list of all memberships.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\MembershipDaoImpl.java

This class provides database access for the membership table.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\SeasonDao.java

An interface for database operations on the season table. The SeasonDao interface provides a method for gaining a list of all seasons.

src\java\fi\cs\helsinki\glindstr\soccerdb\dao\SeasonDaoImpl.java

This class provides database access for the season table.

src\java\fi\cs\helsinki\glindstr\soccerdb\dbconnection\ConnectionProvider.java

This class provides a connection to the specified database.

src\java\fi\cs\helsinki\glindstr\soccerdb\filters\AuthenticationFilter.java

This filter checks that the user is logged in before accessing a page.

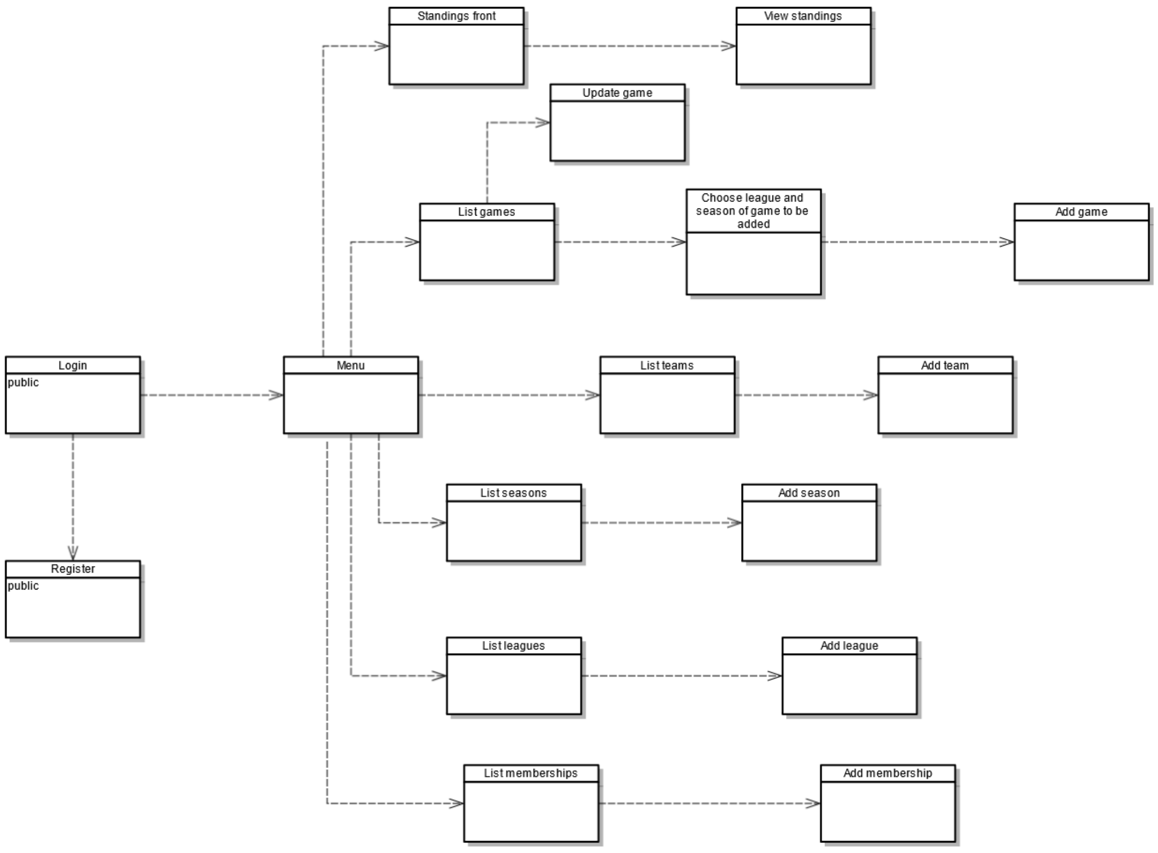
SoccerDb\build\web\WEB-INF\lib\standard.jar and SoccerDb\build\web\WEB-INF\lib\jstl.jar

I used the core tags of the JSP Standard Tag Library (JSTL) to dynamically display content on the jsps.

SoccerDb\web\WEB-INF\web.xml"

Deployment Descriptor. Contains, e.g., servlet and filter mappings.

8 User Interface



9 Installation guide

The application is currently accessible at t-glindstr.users.cs.helsinki.fi/SoccerDb.

Follow these instructions in order to install the application on a tomcat server:

1) Set up a database on the server with the tables specified in this document
2) Download the repository
3) Specify the database driver, server, username and password in the ConnectionProvider class
4) Build a war file
5) Copy the war to the webapps folder of your tomcat installation
6) Start the server and Bob's your uncle

PostgreSQL has been used as the database system. The application may or may not work as is with another database system.

10 User Guide

The application is located at t-glindstr.users.cs.helsinki.fi/SoccerDb. Register a new username or use username: tester password: test123.

11 Testing, Known Bugs and Shortcomings & Ideas for Improvement

I have iteratively tested the application as I have introduced new functionality. I tested that all the use cases worked. I tried to test for user errors, like adding the same team twice, as much as possible.

Input validation for integers and dates only works in browsers supporting HTML5. It is possible two add the same game twice.

Statistics for players would be a great improvement, as well as the ability to execute custom queries. When deleting a record it would be good to show a warning message to prevent accidental deleting, especially since deleting a record can affect many tables. Moreover, having different user groups would be nice, so that not all users could add and delete records.

12 My Experience

This course has been a great learning experience for me. I learned about HTML forms, JSPs, servlets, database management, web servers, MVC and much more. Getting started was the hardest and most time consuming part as the information on the course pages is wildly scattered, partly dated and often confusing. Still, having cleared the initial hurdles the course was enjoyable.

13 Attachments

The SQL statements and Java Docs can be found in the sql and javadoc folders of the GitHub repository, respectively.