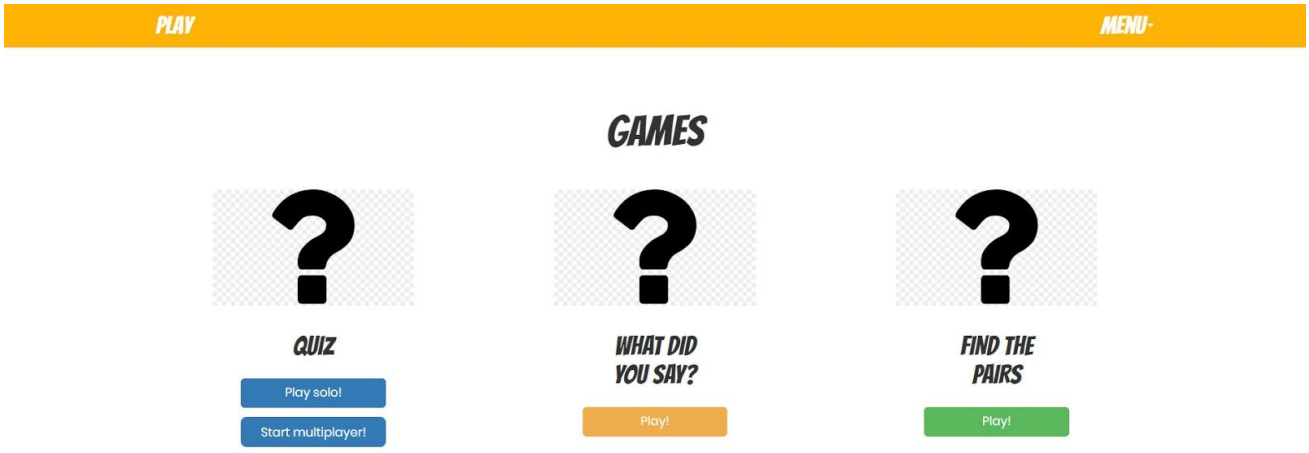
DUOLINGO GAMES

A web application



ABOUT

Duolingo Games is a web application that gamifies the Duolingo language learning experience. It utilizes all the known words in different languages that a user has, and uses them as content in different games, for example quiz (single- and multiplayer), listen-and-type, and memory. When finishing a game the user gets a score, and thereby can compete against all other users in the high score tables. A user can also collect different achievements connected to both game play and simply being a user of the web application itself.

USE CASES

The web application has 11 different defined use cases. Detailed information can be found below.

User logs in
When the user visits the index page, the system requires the user to enter their username and password. The system validates the username and password by sending a validation request to Duolingo. If Duolingo validates the username and password, the user is logged in. Otherwise the system reloads the page and displays an error message.

User can see information on user profile
A user can visit their user profile to see information about their achievements, stats from game-play, stats about their current language, and all their know languages.

User can switch active language
When visiting their user profile, a user can click on a flag to switch to that specific language. When a user switches language, the games will then be played in that language.

User can collect achievements
A user can collect achievements from both game play and simply being a user on the web application. For example, a user can get an achievement for being on of the first 100 users. An example of a gameplay achievement is the *Quick Clicker,* which gives the user an achievement if the user finished a game in under 20 seconds.

User can pick which game to play
By visiting the play page, the user can pick between 3 games, where one could be played both as single player and multiplayer.

User can play Find the pairs
The user can choose to play a memory game by choosing *Find the pairs*. The user is then presented with 6 pairs of words that have been shuffled and placed behind cards. The user opens two cards, if they match they are kept open and the user continues. If they do not match, the cards are automatically turned back around. When the user have found all the pairs, the user is redirected to the score page.

User can play What did you say?
The user can choose to play a listen-and-write type of game by choosing *What did you say?* game. The user is the presented with a game designed as a wizard, and on each page they are presented with a sound clip they can listen to and an input in which they write the word they heard. For each correct word, the user gets 1 point. When the user has finished the wizard, the user is redirected to the score page.

User can play single player Quiz
The user can choose to play single player quiz by choosing *Play solo!* under Quiz. The user is the presented with a game designed as a wizard, and on each page they are presented with a word and three or four possible translations. For each correct translation chosen, the user gets 1 point. When the user has finished the wizard, the user is redirected to the score page.

User can play multiplayer Quiz
No game is fun without being able to play it against friends, and the multiplayer mode for Quiz offers just that. By clicking on the multiplayer button you are asked to give an id to your game *(a default randomly generated one is suggested for you)* and you can give this identifier to anyone you wish to play against given that they are studying the same language as you **the owner** are, and the difference in known words is max 50 *Note this is disabled for the demo, just to facilitate the process.*

When inside the Quiz game, you can see in real time the progress your friends are making and how many wrong/right answer they have got.

Upon finishing the game, you are shown the score page along with your ranking in that game.

User gets a score after a game
When the user has finished a game, the user is redirected to the score page. This page gives the user information about how fast they finished the game and their score. If the game is either Quiz or What did you say?, the user will also see how many questions they had right out of the total possible.

User can view high scores for different games

The user can visit the high score page to see high score charts for the different games. The charts show 10 entries per game, and one entry has the user name, score, and time.

ARCHITECTURE

Duolingo games has a Java EE back-end with a JSF front-end.

BACKEND

Since this is an application for games, the choice was taken to have a model-view-controller structure in the back-end. This is realized in a way where the model has all of the game logic along with an interface and an abstract class for games, which provides easy extendability. The view contains classes known as *backing beans*. These classes are made to hold all the data that is presented to the user, and basically only contains getters and setters. Lastly, the controller has a collection of managed beans. One bean for every created game, and they have methods that are connected to different events in the game; such as starting a new game, or validating if the given answer is correct.

The back end also contains more classes and packages than described above. More details about the complete back-end structure will now follow.

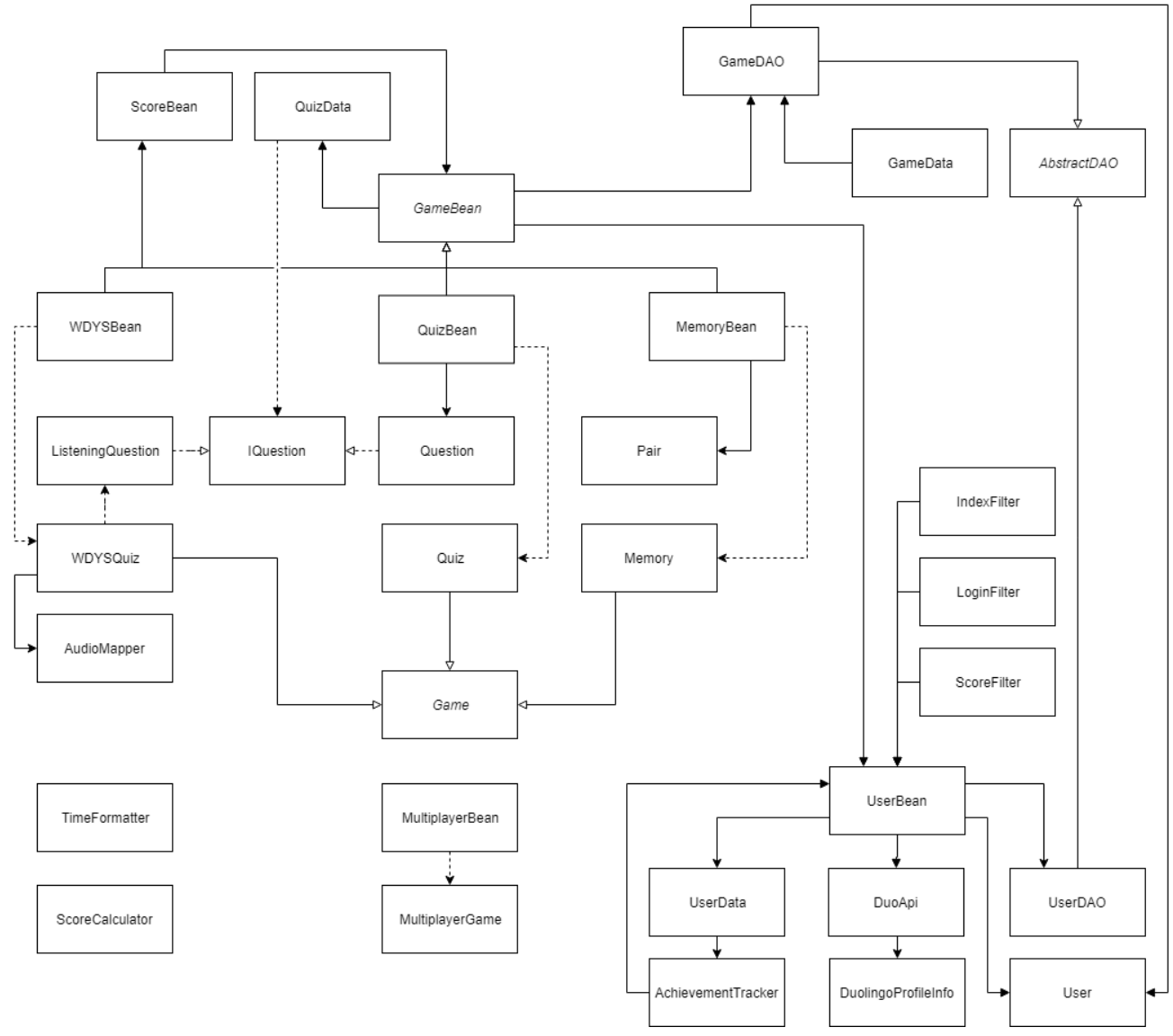
control Contains beans for the different games, user, and score. The game beans all extend the superclass *GameBean*.
database.dao Data Access Objects. These classes are used for communicating with the database. The application creates entries in as well as reads from the database, but does not perform any update or delete operations as they are not needed. **database.entity** The entity classes represents the entries in the database.
filter These classes are responsible for the blocking of non-authorized content as well as blocking the ability to show the score if the user doesn't have a finished game.
model Has general classes and an interface to build specific game classes upon. The model has three child packages; listening, memory, and quiz, which each contains classes with logic for each of the games. Components in the model package are completely independent of the framework(s) we use.
services The classes in the services package are handling the data received from the Duolingo API along with a representative data class to encapsulate main information.
utils Helper classes for formatting time, calculating score, and handles the achievements awarded to a user based on certain criteria (e.g. play time, perfect score).
view Backing beans that holds data that will be presented to the user in the xhtml.

FRONTEND

As previously mentioned, JSF is used for the front end and implemented with the framework Bootsfaces. There has been limited amount of ordinary HTML code, but that has been used for example to show an audio player which Bootsfaces does not have any support for at the moment. Most of the standard Bootstrap CSS is used apart from some customizations.

Javascript has been kept to a minimum since it has simply not been needed to a great extent, but it is used for the multiplayer mode in Quiz as well as in Memory. This is due to the nature of both of these games, where the content on the page needs to be updated without the page being reloaded. In the memory game, the Javascript is communicating with the corresponding game bean in the back-end by using the *remoteCommand* in Bootsfaces. It also receives a return value which it then uses to continue the game. It would be possible to code the whole game only in Javascript, but this would not be the best solution since the user easily could cheat by looking at the source code. By keeping all the important data,such as which two words make up a pair, in the backend this is not an issue.

UML DIAGRAM



FLOW CHART (IN TEXT FORM)

To illustrate how the different components of our project interact, we choose to describe the flow of the process of logging in.

- **INDEX.XHTML**
- The contents of index.xhtml are rendered
- The session scoped bean UserBean is injected onto the page
- If the user has already logged in, run the method **redirect** inside UserBean
- The user writes in her account information i.e. username and password
- By clicking on the button *Login*, a request is made against UserBean's **signin** method and the values entered inside the input boxes are passed through to the bean.
    - **USERBEAN.signin()**
    - Performs usual checks to ensure that the username and password are valid i.e. not empty or null.
    - Creates an instance of **DuoAPI** passing the received username and password to its constructor.
        - **DuoAPI Constructor**
        - Sends an HTTP request against Duolingo's endpoint
        - Populates the newly created object with the information retrieved.
        - Checks the returned token to tell whether the authentication was successful.
    - If the authentication was successful, run **redirect** otherwise run **wrongpass**

DATABASE STRUCTURE

The web application is connected to a PostgreSQL database hosted on Amazon Web Services. An SQL database was chosen for the project due to the group members having previous experience with it, and that SQL's functionality was a good fit for the type of data that was expected to be stored. The database and its schema was designed and created first, and the entity classes were generated from it, rather than the other way around.

The database consists of two tables:

Users	**Gamesession**
- id SERIAL PRIMARY KEY - username VARCHAR(30) NOT NULL	- id SERIAL NOT NULL PRIMARY KEY - time NUMERIC NOT NULL - score INTEGER NOT NULL - userid INTEGER NOT NULL REFERENCES Users(id) - type VARCHAR(4) NOT NULL

TECH STACK

The tech stack of the project is outlined below, split into the aforementioned categories:

Backend:	**Frontend:**	**External resources:**
- Payara/Glassfish server - Java EE - JPA - PostgreSQL - Amazon Web Services	- Bootsfaces - JQuery - Javascript - HTML5 - Custom CSS	- Duolingo's unofficial API

THE TEAM

From left: Niclas Johansson, Abdullatif AlShriaf and Kristina Markan (before using js hence the smiling)