Zirkus Empathico Profile + Database

Database Design

The ZE Database is designed to track changes to the personal score over time. Each Emotion and score has its own table, and every entry represents a change with a timestamp. For future applications, these timestamps and the changes over time can be used to plot progress. Furthermore, the database is designed to keep redundancy at a minimum, with scores not changing not being logged.

Base Table

This table saves the profile data needed.

emailAddress (unique)	emailConfirmed	hashedPassword	gamesPlayed
User.name1@mail.net	true	=)dDp8Bad”D3%	8

Correlation Table

This table creates a correlation between a device ID and a user playing. It is possible to have multiple users on one device, and one user using multiple devices, so the correlation can be ambiguous.

DeviceID	emailAddress
2abd37-2ff3-6a3c-5c321f	user.name1@mail.net
1f3f32-6ac4-7c01-02b383	user.name2@mail.net
2f3312-4fe3-1231-6cbd32	user.name1@mail.net

Score Tables

Every score or value that is changing over time has an own table, so that when only few values are changing, only those changes are actually logged. This also helps e.g. plotting the scores of emotions over time for further applications.

The function updating the scores is designed, so that it only updates the tables it has been given a value for, NULL indicates that the value is not to be updated. For retrieving the data, the retrieving function collects every tables value and sends it back with the timestamp of the newest entry. Every table has their own Unique Entry-IDs. For the sake of simplicity only the different column of every table is shown.

Same for every table			Score columns			
Entry-ID (unique)	**emailAddr**	**Timestamp**	**ELO**	**K-Value**	**Sad**	**…**
1	user.name1@mail.net	1559974650	1759		789	
2	user.name2@mail.net	1559974880	859			510
3	user.name1@mail.net	1559976356	1785	0.8		

FUNCTIONS

To operate on the database, following functions are needed:

- **createUser(String emailAddr, String Password): boolean success**
  - This function takes an Email-Adress and a password as arguments, and creates all the necessary database entries and initializes default values for the new user. It returns whether or not the operation was a success.
- **deleteUser(String emailAddr, String Password): boolean success**
  - After verification of username and password (verifyPassword) this function deletes every database table entry correlated to the given Email-Adress, deleting the whole user profile.
- **hashPassword(String Password): String hashedPassword**
  - A given password is hashed and returned.
- **verifyPassword(String emailAddr, String Password): boolean success**
  - The given Password is hashed, and the function looks up if the combination of the given Email-Address and the hashed Password exists in the base table. It returns the answer as a boolean value. This function is called upon login and deletion of profile.
- **retrieveData(String emailAddr): jsonObject DBdata**
  - The function returns a jsonObject filled with the current scores and a timestamp. It retrieves the latest scores (highest Entry-ID) of the given emailAddress in every table and attaches the timestamp of the newest entry.
- **pushData(jsonObject DBdata): boolean success**
  - This function takes the values from the given jsonObject that are NOT NULL and updates the values in their respective tables.
- **logoutUser( ): boolean success**
  - sets the environment so that no more data can be pushed or retrieved to/from the database.
- **getCurrentUser( ): String emailAddr**
  - Returns the email of the currently logged in User
- **getDeviceID( ): String deviceID**
  - Returns the ID of the current device
- **correlateDevice(String emailAddr, String deviceID): boolean success**
  - Creates an entry in the correlation table for the given device and user Email