Webservice Definition

Url:

**Methode**: register

**Parameter**: String Name, String Email, String Password, String Type

**Return**: int userID

**Beschreibung**: legt einen neuen Account mit den mitgegebenen Parametern an in der Tabelle Veranstalter oder Teilnehmer

**SQL-Anweisung**: insert into Account values (seq\_account\_id.NEXTVAL, Name, Email, Password);  
select MAX(a.id) into actualID from Account a;  
 insert into Teilnehmer values (actualID, 0);  
insert into Veranstalter values (actualID);

**Angewendet in Activity:**

**Url**:

**Methode:** login

**Parameter:** String email, String password

**Return:** int userID

**Beschreibung:** die Datenbank wird durchsucht ob der User existiert und die Id wird zurückgegeben

**SQL-Anweisung:** select \* from Account a where a.email = email AND a.password = password;

**Angewendet in Activity:**

**Url**:

**Methode**: getUser

**Parameter**: int userID

**Return**: String user

**Beschreibung**: gibt die vollständigen Userdaten zurück

**SQL-Anweisung**: select …

**Angewendet in Activity**:

Alle changes werden in einem Webservice implementiert

Paramenter: JSON-Objekt mit dem ganzen Account

**Url**:

**Methode**: changePassword

**Parameter**: int userID, String oldPassword, String newPassword

**Return**: Boolean success

**Beschreibung**: Ändert das Passwort des Users und gibt zurück ob dies erfolgreich war

**SQL-Anweisung**: update …

**Angewendet in Activity:**

**Url**:

**Methode**: changeName

**Parameter**: int userID, String oldName, String newName

**Return**: Boolean success

**Beschreibung**: Ändert den Namen des Users und gibt zurück ob dies erfolgreich war

**SQL-Anweisung:** update …

**Angewendet in Activity:**

**Url**:

**Methode**: changeEmail

**Parameter**: int userID, String oldEmail, String newEmail

**Return**: Boolean success

**Beschreibung**: Ändert die E-Mail des Users und gibt zurück ob dies erfolgreich war

**SQL-Anweisung**: update …

**Angewendet in Activity:**

**Url**:

**Methode**: deleteUser

**Parameter**: int userID

**Return**: Boolean success

**Beschreibung**: prüft innerhalb der Methode ob der User ein Teilnehmer ist oder ein Admin und ruft je  
 nach dem die Methode deleteParticipationsOfParticipant oder deleteEventsFromAdmin   
 auf und löscht bei Erfolg den User.

**SQL-Anweisung:** delete…

**Angewendet in Activity:**

**Url**:

**Methode**: deleteParticipationsOfParticipant

**Parameter**: int userID

**Return**: Boolean success

**Beschreibung**: löscht alle Teilnahmen an Events von einem Teilnehmer.

**SQL-Anweisung**: delete

**Angewendet in Activity:**

**Url**:

**Methode**: deleteEventsFromAdmin

**Parameter**: int userID

**Return**: Boolean success

**Beschreibung**: ruft die Methode informAllUsersAboutChange mit der Nachricht, dass diese Veranstaltung abgesagt wurde auf. Danach werden alle Teilnahmen von diesem Event entfernt.

**SQL-Anweisung**: delete

**Angewendet in Activity:**

**Url**:

**Methode**: notifyAllUserAboutChange

**Parameter**: int veranstaltungsID, String message

**Return**: Boolean success

**Beschreibung**: Geht alle dieser Veranstaltung durch und ruft für jeden User die Methode informUserAboutChange auf.

**SQL-Anweisung**: select

**Angewendet in Activity:**

**Url**:

**Methode**: notifyUserAboutChange

**Parameter**: int userID, String message

**Return**: Boolean success

**Beschreibung**: Informiert den aktuellen User über eine Änderung.

**SQL-Anweisung**: select

**Angewendet in Activity:**

**Url**:

**Methode**: AllEventsBetween

**Parameter**: Datetime start, Datetime end

**Return**: List<Veranstaltungen> veranstaltungen

**Beschreibung**: gibt alle Veranstaltungen die im gegebenen Zeitraum stattfinden zurück.

**SQL-Anweisung**: select

**Angewendet in Activity:**

**Url**:

**Methode**: EventsWhereParticipantWillTakePart

**Parameter**: int userID

**Return**: List<Veranstaltungen> veranstaltungen

**Beschreibung**: Gibt alle Veranstaltungen bei denen der Teilnehmer sich angemeldet hat zurück.

**SQL-Anweisung**: select \* from Veranstaltung v where exists (select id\_veranstaltung from teilnahmen where v.id = t.id and Trunc(datetime) > = Trunc(sysdate))

**Angewendet in Activity:**

**Url**:

**Methode**: EventsWhereParticipantTookPart

**Parameter**: int userID

**Return**: List<Veranstaltungen> veranstaltungen

**Beschreibung**: Liefert alle Veranstaltungen bei denen der Teilnehmer teilgenommen hat zurück

**SQL-Anweisung**: select \* from Veranstaltung v where exists (select id\_veranstaltung from teilnahmen where v.id = t.id and Trunc(datetime) <= Trunc(sysdate))

**Angewendet in Activity:**

**Url**:

**Methode**: getEvent

**Parameter**: int veranstaltungsID

**Return**: Veranstaltung veranstaltung

**Beschreibung**: Gibt eine Veranstaltung zurück

**SQL-Anweisung**: select \* from Veranstaltung where id = veranstaltungsID;

**Angewendet in Activity:**

**Url**:

**Methode**: registerToEvent

**Parameter**: int userID, int eventID

**Return**: Boolean successful

**Beschreibung**: Meldet den Teilnehmer bei einem Event an, welches in der Zukunft stattfinden wird

**SQL-Anweisung**: insert into teilnahme….

**Angewendet in Activity:**

**Url**:

**Methode**: unsubscribeFromEvent

**Parameter**: int userID, int eventID

**Return**: boolen success

**Beschreibung**: Entfernt die Teilnahme eines Users von einer Veranstaltung

**SQL-Anweisung**: delete…

**Angewendet in Activity:**

**Url**:

**Methode**: generateEvent

**Parameter**: String name, int userID, String location, LocalDate date, int maxTeilnehmer, int minTeilnehmer, String details, String Sportart

**Return**: Veranstaltung veranstaltung

**Beschreibung**: Generiert eine neue Veranstaltung und speichert diese in der Datenbank

**SQL-Anweisung**: insert into Veranstaltung

**Angewendet in Activity:**

**Url**:

**Methode**: AllEventsFromAdmin

**Parameter**: int userID

**Return**: List<Veranstaltungen> veranstaltungen

**Beschreibung**: gibt alle Veranstaltungen eines Veranstalters zurück

**SQL-Anweisung**: select \* from Veranstaltung where id\_veranstalter = userID;

**Angewendet in Activity:**

**Url**:

**Methode**: changeEvent

**Parameter**: int eventide, String name, int userID, String location, LocalDate date, int maxTeilnehmer, int minTeilnehmer, String details, String Sportart

**Return**: Veranstaltung changedVeranstaltung

**Beschreibung**: Ändert die Daten einer Veranstaltung und gibt ruft die Methode notifyAllUserAboutChange auf

**SQL-Anweisung**: update…

**Angewendet in Activity:**