Specification - Sports Club App

The goal is to create a simple application that would be helpful in the management of a local sports club. In the app, every player will be able to sign up for training, and matches, also he will be able to see an overview of training and matches where he is and was signed up. Also, the player will see in the app the information about the payment of the membership fee after successful registration and the app will offer the possibility to generate a QR code for payment. Only the user who is allowed by the coach will be able to register in the app. Additionally, the coach will be able to add and change practices and games, mark paid memberships, and have an overall view of all players' participation statistics.

Functional requirements

Roles

- within the application:

● Player - club player. Can log in/out of training sessions and matches. He will see an overview of training sessions and matches where he is logged in. Will see information about membership payment (paid/unpaid) and will be able to generate a QR code for payment in case of unpaid membership.

● Coach - coach of the club. He will be able to add/remove training sessions and matches, and also change their times. He will be able to send notifications to players if they have not yet expressed their participation. He will have an overview of participation statistics. Mark paid memberships to individual players.

- at the database level:

● Application admin (me) - will change the roles of players/coaches in agreement with the club. In agreement with the coach, will only allow selected people to register, so that not just anyone can register for the club. Change the payment status of the membership fee (after the coach informs him).

Use Case diagram = TO DO

Player
- registration
- log in/log out
- sign in/out of training and matches
- overview of training/matches
- statement of payment of the membership fee

Coach
- add trainings/matches
- change training/match times
- notification to players
- overview of statistics of all players

Admin
- changes in the database after agreement with the club and coach

Data model

Roles

- list of role types (player/coach)
- *role_id* = role identifier
- *role* = role name

Events_type

- list of event types (training/match)
- *type* = event identifier
- *name* = event name

Teams

- list of teams
- *team_id* = team identifier
- *team* = team name

Users

- list of specific application users
- *user_id* = user identifier
- *role_id* = the role of the user (from table Roles)
- *team_id* = the user's team (from table Teams)
- *membership_fee_status* = payment status of membership fee (paid/unpaid)
- *name*
- *surname*
- *birth*
- *email*

Events

- list of specific created events
- *event_id* = identifier of the created event
- *type* = type of the event (from table Events_type)
- *team* = team that created the event (from table Teams)
- *name* = name of the event
- *location* = location of the event
- *date*

Events_Users

- list of users signed up for the event
- *event_id* = identifier of the event (from table Events)
- *user_id* = identifier of the user (from table Users)
- *response* = response of the user (participate/not participate)

Accounts

- list of people who can register for a specific team (records added by the administrator in agreement with the coach). Only people in this table will be allowed to register.
- *email*

- *password* = contains hash values
- is_registrated
- user_id = identifier of the user (from table Users), NULL at the beginning

ER diagram = TO DO

Architecture

The application will be based on the client-server architecture and use PWA (Progressive Web App), so we can send the notifications straight to mobile phones.

Technological requirements

- Client-side: JavaScript, HTML5, CSS3
- Server-side: Ruby on Rails (framework), Ruby
- Database: SQLite3
- Interface client - server: Rest API
- Hosting: TO DO
- Supported browsers: Chrome, Firefox, Safari