FSB Doku Rebuilt out of all possible Channels

* HOST : 188.166.166.49
* PORT : 8580

Technologies Used

* Hapi.jsA rich framework for building applications and services. hapi enables developers to focus on writing reusable application logic instead of spending time building infrastructure.
* Knex.js  Knex.js is a "batteries included" SQL query builder for Postgres, MSSQL, MySQL,MariaDB, SQLite3, and Oracle designed to be flexible, portable, and fun to use. It features both traditional node style [callbacks](http://knexjs.org/#Interfaces-Callbacks) as well as a [promise](http://knexjs.org/#Interfaces-Promises) interface for cleaner async flow control, [a stream interface](http://knexjs.org/#Interfaces-Streams), full featured [query](http://knexjs.org/#Builder) and [schema](http://knexjs.org/#Schema) builders,[transaction support (with savepoints)](http://knexjs.org/#Transactions), connection [pooling](http://knexjs.org/" \l "Installation-pooling)and standardized responses between different query clients and dialects.
* Bookshelf.js Bookshelf is a JavaScript ORM for Node.js, built on the [Knex](http://knexjs.org/) SQL query builder. Featuring both promise based and traditional callback interfaces, providing transaction support, eager/nested-eager relation loading, polymorphic associations, and support for one-to-one, one-to-many, and many-to-many relations.
* Postgressql (pronounced "post-gress-Q-L") is an open source relational database management system ( DBMS ) developed by a worldwide team of volunteers.PostgreSQL is not controlled by any corporation or other private entity and the source code is available free of charge.
* Forever A simple CLI tool for ensuring that a given script runs continuously

Directory Structure

1. App
   1. Controllers
   2. Models
   3. Helper.js
   4. Routes.js
   5. Auth.js
2. Bootstrap
   1. Providers.js
   2. Start.js
3. Config
   1. Database.js
4. Migrations
5. Resources
   1. Password
6. Seeds
   1. Games.js
   2. Sports.js
   3. Users.js
7. .env
8. Index.js
9. Knexfile.js
10. Package.json

**Notice : for install all dependecies run npm install**

Install DB Schema

| Code Block |
| --- |
| |  |  | | --- | --- | | language | bash | | title | Install DB Schema | |
| $ npm install knex -g  $ knex migrate:latest |

Authentication

Set Header Authorization :

| Code Block |
| --- |
| |  |  | | --- | --- | | language | bash | | title | Auth Header | |
| Barear {token from signin & signup} |

USER

|  |  |  |  |
| --- | --- | --- | --- |
| **REQUEST** | **URL** | **DESCRiPTION** | **PARAMS** |
| GET | /user | Getting all users saved in DB |  |
| POST | /user/:user | Creating a User in System | {User} user - User Object as Json |
| GET | /user/:userid | Getting specified User | {int} userid - The Id of the User |
| PUT | /user/:userid | Updating a specified User | {int} userid - The Id of the User  {User} user - User Object as Json |
| DELETE | /user/:userid | Deleting the specified User | {int} userid - The Id of the User |
| GET | /userbygeo/:lat/:lng/:radius | Getting all users in specified Geo Koos | {float} lat - The Latitude of the Geo Koos  {float} lng - The Longitude of the Geo Koos  {int} radius - The Radius of the Geo Koos |

GAME

|  |  |  |  |
| --- | --- | --- | --- |
| **REQUEST** | **URL** | **DESCRIPTION** | **PARAMS** |
| GET | /game | Getting all games saved in DB |  |
| POST | /game/:game | Creating a Game in System | {Game} game - Game Object as Json |
| GET | /game/:gameid | Getting specified Game | {int} gameid - The Id of the game |
| DELETE | /game/:gameid | Deleting the specified Game | {int} gameid - The Id of the game |
| PUT | /game/:gameid | Updating a specified Game | {int} gameid - The Id of the game  {Game} game - Game Object as Json |
| GET | /gamebygeo/:lat/:lng/:radius | Getting all Games in specified Geo Koos | {float} lat - The Latitude of the Geo Koos  {float} lng - The Longitude of the Geo Koos  {int} radius - The Radius of the Geo Koos |
| GET | /gamesonmap/:lat/:lng/:radius/:starttimefrom/:starttimeto/:sportid | Gettng all Games on Map with specified attributes | {float} lat - The Latitude of the Geo Koos  {float} lng - The Longitude of the Geo Koos  {int} radius - The Radius of the Geo Koos  {Date} starttimefrom - The Date where the Period of the starttime begins  {Date} starttimeto - The Date where the Period of the starttime ends  {int} sportid - The Id of the sport the game should have |
| GET | /gamebycreator/:creatorid | Getting the Games a User has created | {int} creatorid - The Id of the Creater(User) of the Game |
| GET | /gamebyparticipant/:participantid | Getting the Games a User has or is participating | {int} participantid - The Id of a Particpipant |
| GET | /gamebyweekday/:weekday | Getting all games for a specified Weekday | {Date} weekday - The Weekday for searched Games (taken from Date.getDay()) |
| GET | /gamebyweek/:week | Getting all games for a specified Week | {Date} week - The Week for searched Games (taken from Date.getWeek()) |
| GET | /gamebydaypart/:daypart | Getting all games for a specified part of day (morning, afternoon, evening) | {string} daypart - The Daypart for searched Games (morning, afternoon, evening) |

LOCATION

|  |  |  |  |
| --- | --- | --- | --- |
| **REQUEST** | **URL** | **DESCRIPTION** | **PARAMS** |
| GET | /location | Getting all Locations saved in DB |  |
| GET | /location/:locationid | Getting specified location | {int} locationid - The Id of the Location |
| PUT | /location/:locationid | Updating a specified Game | {int} locationid - The Id of the Location  {Location} location - The Data of the Location Update |
| POST | /location/:location | Creating a Location in System | {Location} location - The Location which will be created |
| DELETE | /location/:locationid | Deleting the specified Location | {int} locationid - The Id of the Location |
| GET | /locationbygeo/:lat/:lng/:radius | Getting all Locations in specified Geo Koos | {float} lat - The Latitude of the Geo Koos  {float} lng - The Longitude of the Geo Koos  {int} radius - The Radius of the Geo Koos |

NOTIFICATION

|  |  |  |  |
| --- | --- | --- | --- |
| **REQUEST** | **URL** | **DESCRIPTION** | **PARAMS** |
| GET | /notification/:notificationid | Getting specified Notification | {int} notificationid - The Id of the Notification |
| GET | /notification | Getting all Notifications saved in DB |  |
| POST | /notification/ | Creating a Notification in System | {Notification} notification - The Notification which will be created |
| PUT | /notification/:notificationid | Updating a specified Notification | {int} notificationid - The Id of the Notification  {Notification} notification - The Data of the Notification Update |
| GET | /notificationbyuser/:userid | Getting the Notifications of a user | {int} userid - The Id of the User |
| DELETE | /notification/:notificationid | Deleting the specified Notification | {int} notificationid - The Id of the Notification which will be deleted |

...

|  |  |  |  |
| --- | --- | --- | --- |
| **REQUEST** | **URL** | **DESCRIPTION** | **PARAMS** |
| GET | /profile/:profileid | Getting specified Profile | {int} profileid - The Id of the Profile |
| POST | /profile/ | Creating a Profile in System | {Profile} profile - The Profile which will be created |
| GET | /profile | Getting all Profiles saved in DB |  |
| DELETE | /profile/:profileid | Deleting the specified Profile | {int} profileid - The Id of the Profile which will be deleted |
| PUT | /profile/:profileid | Updating a specified Profile | {int} profileid - The Id of the Profile  {Profile} profile - The Data of the Profile Update |
| POST | /profilebyuser/:userid | Getting the Profile of specified User | {int} userid - The Id of the User |
| POST | /upload | Uploading a Avatar on Server | {string} profile - Name where in which folder the picture will be stored  {stream} picture - The picture which will be uploaded |
| GET | /pic/:profile/:filename | Getting a specified Pic from Server as a Stream | {string} profile - The String in which folder will be saved  {string} filename - The Filename of searched file |
| GET | /picsforprofile/:profile | Getting a List of Filepaths of the pics for a profile | {string} profile - The Profile of the pics |

SPORT

|  |  |  |  |
| --- | --- | --- | --- |
| **REQUEST** | **URL** | **DESCRIPTION** | **PARAMS** |
| GET | /sport/:sportid | Getting specified Sport | {int} sportid - The Id of the Sport |
| POST | /sport/ | Creating a Sport in System | {Sport} sport - The Date of the Sport which will be created |
| GET | /sport | Getting all Sports saved in DB |  |
| PUT | /sport/:sportid | Updating a specified Sport | {int} sportid - The Id of the Sport  {Sport} sport - The Data of the Sport which will be updated |
| DELETE | /sport/:sportid | Deleting the specified Sport | {int} sportid - The Id of the Sport which will be deleted |
| POST | /sportsbyids | Finding a Collection of Sports by their ids | sportids - array of searched Sport Ids |
| POST | /sportsbyuser | Finding a Collection of Sport related to the User | {User} user - The User Object which Sports are searched for |

BUDDYCONNECTION

|  |  |  |  |
| --- | --- | --- | --- |
| **REQUEST** | **URL** | **DESCRIPTION** | **PARAMS** |
| GET | /buddyconnection/:connectionid | Getting specified BuddyConnection | {int} connectionid - The Id of the BuddyConnection |
| POST | /buddyconnection | Creating a BuddyConnection in System | {BuddyConnection} connection - The Date of the BuddyConnection which will be created |
| GET | /buddyconnection | Getting all BuddyConnections saved in DB |  |
| PUT | /buddyconnection/:connectionid | Updating a specified BuddyConnection | {int} connectionid - The Id of the BuddyConnection  {BuddyConnection} connection - The Data of the BuddyConnection which will be updated |
| DELETE | /buddyconnection/:connectionid | Deleting the specified BuddyConnection | {int} connectionid - The Id of the BuddyConnection which will be deleted |
| GET | /connectionbyuser/:userid | Finding a Collection of Connections related to the User | {int} useris - The Id of the User which Connections are searched for |

AUTH

|  |  |  |  |
| --- | --- | --- | --- |
| **REQUEST** | **URL** | **DESCRIPTION** | **PARAMS** |
| GET | /logout | Logout of the User |  |
| GET | /login | the View after Login |  |
| POST | /signupuser | Registration a User and storing it | {string} email - The email of User which will be created  {string} password - The password of User which will be created |
| POST | /login | Login a User | {doclet} doclet which will be processed |
|  |  |  |  |

...

Later Email contained this (as a change by Otti)

**Notice : for install all dependecies run**

| Code Block |
| --- |
| |  |  | | --- | --- | | language | bash | | title | install node packages | |
| $ npm install |

## Install DB Schema

| Code Block |
| --- |
| |  |  | | --- | --- | | language | bash | | title | Install DB Schema | |
| $ npm install knex -g  $ knex migrate:latest |

...

| Code Block |
| --- |
| |  |  | | --- | --- | | language | bash | | title | Auth Header | |
| Barear {token from signin & signup} |

## ~~USER~~

~~{float} lat - The Latitude of the Geo Koos~~

~~{float} lng - The Longitude of the Geo Koos~~

~~{int} radius - The Radius of the Geo Koos~~

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **REQUEST** | **URL** | **DESCRiPTION** | **PARAMS** | ~~GET~~ | ~~/user~~ | ~~Getting all users saved in DB~~ | **Auth required** |
| POST | /api/v1/signup |  |  |  |  |  |  |
| POST | /api/~~user~~v1/~~:user~~ | ~~Creating a User in System~~ | ~~{User} user - User Object as Json~~signin |  |  |  |  |
| POST | /api/v1/forgot-password | Moved to firebase |  |  |  |  |  |
| GET | /  ~~user/:userid~~ | ~~Getting specified User~~ | ~~{int} userid - The Id of the User~~ |  |  |  |  |
| ~~PUT~~ | ~~/user/:userid~~ | ~~Updating a specified User~~ | ~~{int} userid - The Id of the User~~  ~~{User} user - User Object as Json~~ |  |  |  |  |
| ~~DELETE~~ | ~~/user/:userid~~ | ~~Deleting the specified User~~ | ~~{int} userid - The Id of the User~~ |  |  |  |  |
| ~~GET~~ | ~~/userbygeo/:lat/:lng/:radius~~ | ~~Getting all users in specified Geo Koos~~ | password\_reminder/{token} | Moved to firebase |  |  |  |
| POST | /password\_reminder/{token} | Moved to firebase |  |  |  |  |  |
| GET | /api/v1/users/{user\_id} |  |  |  |  |  |  |
| PUT | /api/v1/users/{user\_id} |  |  |  |  |  |  |
| GET | /api/v1/users/{user\_id}/sports |  |  |  |  |  |  |
| POST | /api/v1/users/{user\_id}/sports/{sport\_id} |  |  |  |  |  |  |
| DELETE | /api/v1/users/{user\_id}/sports/{sport\_id} |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

## GAME

|  |  |  |  |
| --- | --- | --- | --- |
| **REQUEST** | **URL** | **DESCRIPTION** | **PARAMS** |
| GET | /game | Getting all games saved in DB |  |
| POST | /game/:game | Creating a Game in System | {Game} game - Game Object as Json |
| GET | /game/:gameid | Getting specified Game | {int} gameid - The Id of the game |
| DELETE | /game/:gameid | Deleting the specified Game | {int} gameid - The Id of the game |
| PUT | /game/:gameid | Updating a specified Game | {int} gameid - The Id of the game  {Game} game - Game Object as Json |
| GET | /gamebygeo/:lat/:lng/:radius | Getting all Games in specified Geo Koos | {float} lat - The Latitude of the Geo Koos  {float} lng - The Longitude of the Geo Koos  {int} radius - The Radius of the Geo Koos |
| GET | /gamesonmap/:lat/:lng/:radius/:starttimefrom/:starttimeto/:sportid | Gettng all Games on Map with specified attributes | {float} lat - The Latitude of the Geo Koos  {float} lng - The Longitude of the Geo Koos  {int} radius - The Radius of the Geo Koos  {Date} starttimefrom - The Date where the Period of the starttime begins  {Date} starttimeto - The Date where the Period of the starttime ends  {int} sportid - The Id of the sport the game should have |
| GET | /gamebycreator/:creatorid | Getting the Games a User has created | {int} creatorid - The Id of the Creater(User) of the Game |
| GET | /gamebyparticipant/:participantid | Getting the Games a User has or is participating | {int} participantid - The Id of a Particpipant |
| GET | /gamebyweekday/:weekday | Getting all games for a specified Weekday | {Date} weekday - The Weekday for searched Games (taken from Date.getDay()) |
| GET | /gamebyweek/:week | Getting all games for a specified Week | {Date} week - The Week for searched Games (taken from Date.getWeek()) |
| GET | /gamebydaypart/:daypart | Getting all games for a specified part of day (morning, afternoon, evening) | {string} daypart - The Daypart for searched Games (morning, afternoon, evening) |

...

* User is able to add Venues in the FSB Backend by a Webinterface
* PHP Framework Laravel
* Serverpath : /var/www/html/fsb-admin

URL : <http://admin.findsomebuddy.de/back-office>

...

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | |  | | --- | |  | | [NGINX Config](https://petitcode.atlassian.net/wiki/display/FSB/NGINX+Config?src=mail&src.mail.timestamp=1496690417668&src.mail.notification=com.atlassian.confluence.plugins.confluence-notifications-batch-plugin%3Abatching-notification&src.mail.recipient=ff808081525260960152739576050002&src.mail.action=view) | |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | |  | |  | | --- | | **Ottavio Braun** edited this page | | |  |  | | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | | |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | Here's what changed: |  |  |  | | --- | --- | | |  | | --- | | Certs Bundle Certs : /etc/ssl/findsomebuddy.de/[findsomebuddy.de](http://findsomebuddy.de).chained.crt  Key : /etc/ssl/findsomebuddy\_de.key  ...  File : /etc/nginx/sites-available/[findsomebuddy.de](http://findsomebuddy.de) Homepage: server {  listen 80;  server\_name [findsomebuddy.de](http://findsomebuddy.de) [www.findsomebuddy.de](http://www.findsomebuddy.de);  root /var/www/\_[findsomebuddy.de](http://findsomebuddy.de);  ...  include global/common.conf;  include global/wp.conf;  include global/wp-ms.conf; } Admin   server {  listen 80;  server\_name [admin.findsomebuddy.de](http://admin.findsomebuddy.de);  root /var/www/html/fsb-admin/public;  location /templates/ {  alias /var/www/html/fsb-admin/public/templates/$1;  }  # access\_log /etc/nginx/logs/[findsomebuddy.de](http://findsomebuddy.de).access.log main;  error\_log /etc/nginx/logs/[findsomebuddy.de](http://findsomebuddy.de).log error;  # error\_page 404 /404.html;  # error\_page 403 /403.html;  ... | | | | | |