

# Gym Progress Logger

---

*Pawel Paszki (Applied Computing Year 3)*

*Student number: 20063617*

*Component Development*

## Contents

I. Overview .....	3
II. Data Model Design .....	5
a. Class diagram.....	5
b. Sample data in JSON format.....	6
c. Non-trivial issues .....	8
III. UI Design .....	9
a. Login (URL: “/login”) .....	9
b. Home (URL: “/home”).....	12
c. TrainingSessionList (URL: “/users/:id”) .....	15
d. MuscleGroupSessionList (URL: “/trainingsessions/:id”).....	17
e. ExerciseUnitList (URL: “/musclegroupsessions/:id”).....	18
f. ChartGenerator (URL: “/charts/:id”) .....	20
g. MuscleList (URL: “/muscles”) .....	24
h. ExerciseList (URL: “/musclegroupexercises/:id”) .....	27
i. ExerciseInfo (URL: “exerciseinfo/:id”).....	31
IV. Extra Features .....	33
V. Independent Learning .....	33

## I. Overview

The application serves as a keeper of training sessions of users. It could be used by a gym administrator to enter data mentioned above and observe progress of each user by generating charts for each of the exercises in specified dates range.

Home screen displays list of users kept in persistent file. For each user, there is a picture, which is loaded from online URL, button allowing to access “charts” view. Details of each user can be edited (attributes will be mentioned later). It is also possible to delete any user. This will result in removing all data corresponding to the user. List of the users can be displayed in alphabetic order and can also be sorted by bodyweight of users (ShouldComponentUpdate example is shown to prevent re-rendering of filtered users, when there is no change in the rendered list). It is also possible to filter users by typing their name into search box. New user can be added as well.

When user’s name is clicked, list of his/her training sessions is displayed. The name of the session must be in format “dd/mm/yyyy” representing the date of the training session. The textfield for adding new training session is validated and only date in valid format will be accepted to add new session. The date of the session can be edited. Session can also be deleted, which will result in removing all muscle group sessions for this training session along with all exercises corresponding to the muscle group sessions.

After the training session’s date is clicked, list of muscle group sessions is displayed. Each of sessions (by default with its exercises) can be deleted. Session object cannot be edited, because it represents a valid muscle group name. It is also impossible to add muscle group session with name other than one available in the list of muscle groups. Once a muscle group session is added, by default one exercise is also added to it.

Name of the muscle group session is displayed as a hyperlink, which directs to the list of exercises for this session. If there is only one exercise, it cannot be deleted. To add subsequent exercises, buttons representing them can be pressed (this feature is provided to make sure that valid exercises are added and to avoid typing their (sometimes) long names). By default, all editable attributes of exercise are set to 0 (ie weight, number of reps and series).

From home view (or by specifying appropriate URL) “charts” view can be accessed. The hyperlink is parameterized – each item in the user list has its own button. In order to generate chart name of the exercise has to be specified. **Autosuggest** component (see in extras for more

info) was implemented to allow to pick exercise from the predefined list, which is accessed from stubAPI. Starting date and end date also must be specified. Both of them have to be in mentioned earlier format (“dd/mm/yyyy”). “type valid date” info will be displayed if any of the dates is not in desired format. In order to extract data for the chart training sessions, muscle group sessions and exercise units have to be traversed. If number of displayed units in the chart is greater than 8, format of dates in x axis will be trimmed to the shorted possible format (ie 07/08 to 7/8)

In all views, Muscles & Exercises button is visible. When clicked, it directs to the list of muscles, which can be expanded by adding new muscles. Muscle groups can also be removed (or their names edited), once added. There is however safeguard provided which prevents from deleting or renaming 7 predefined constant muscle groups (denoted with attribute constant: true).

Name of each muscle group is also a link to exercises provided for the muscle group. The same safeguard is used in order to prevent from removing predefined exercises. New exercises can be added, edited and removed.

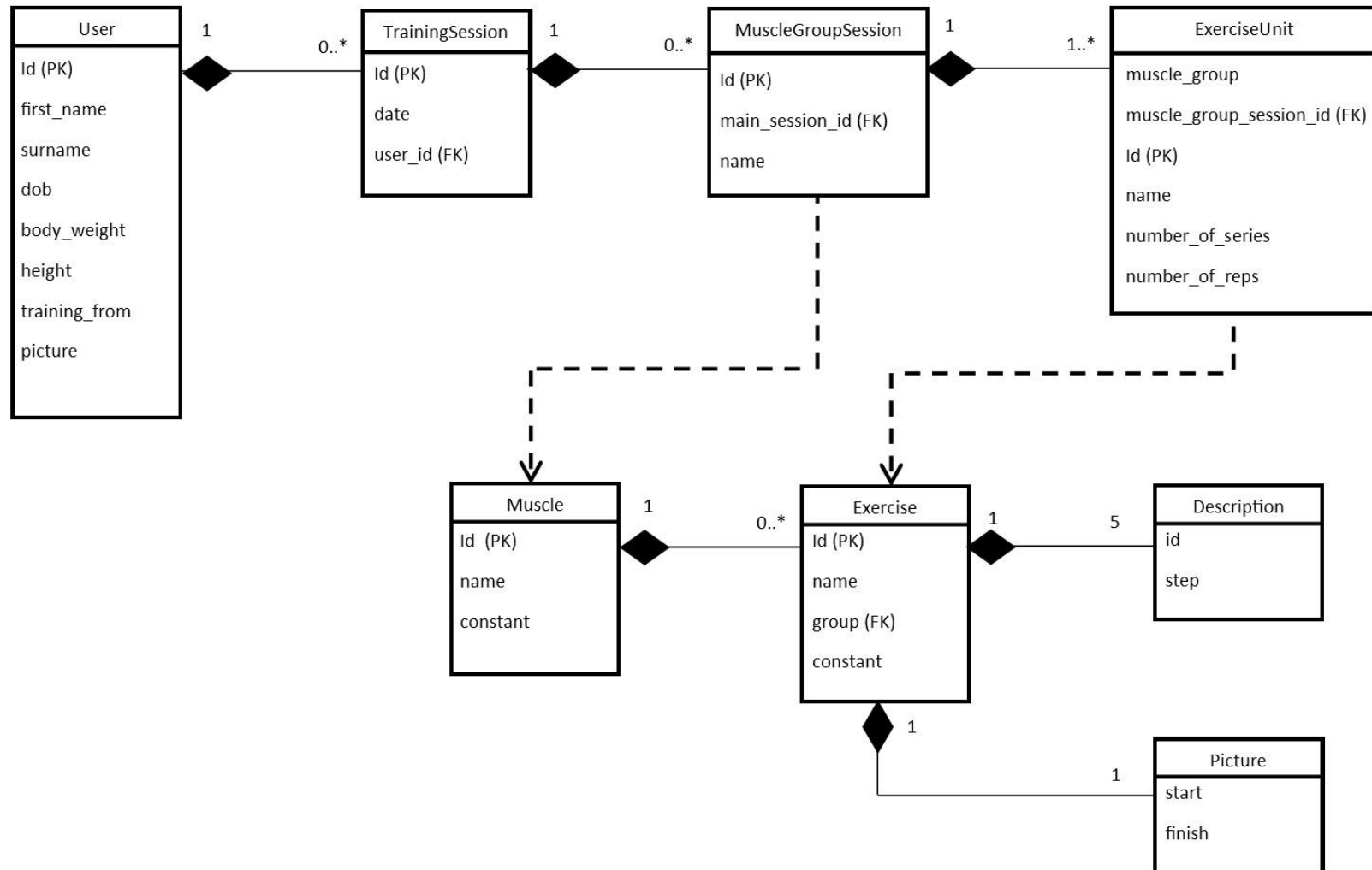
Each of exercises names is a hyperlink to view, which displays the description of the exercise and two pictures showing starting and ending point of the exercise’s performance.

#### General note about forms and adding new items:

When new item is added, it gets id, which is equal to current highest id value from given collection increased by one. Before item is updated or added, simple validation is performed, if in place where number is expected, input is a number, the same for a string and date in specified format.

## II. Data Model Design

### a. Class diagram



## b. Sample data in JSON format

User:

```
{
  "id": 5,
  "first_name": "Darell",
  "surname": "Kent",
  "dob": "10/09/1998",
  "body_weight": 111,
  "height": 191,
  "training_from": "12/09/2013",
  "picture": "https://s17.postimg.org/b55q322in/5.jpg"
}
```

TrainingSession:

```
{
  "id": 0,
  "date": "04/07/2016",
  "user_id": 1
}
```

MuscleGroupSession:

```
{
  "id": 0,
  "main_session_id": 0,
  "name": "chest"
}
```

ExerciseUnit:

```
{
  "muscle_group": "biceps",
  "muscle_group_session_id": 1,
  "id": 3,
  "name": "barbell curl",
  "weight": 60,
  "number_of_series": 4,
  "number_of_reps": 6
}
```

Muscle:

```
{
  "id": 1,
  "constant": true,
  "name": "chest"
}
```

Exercise:

```
{
  "id": 21,
  "constant": true,
  "name": "t-bar row",
  "group": "back",
  "descriptions": [
    {
      "id": 1,
      "step": "Load the bar to an appropriate weight for your training. Stand astride the bar and bend at the hips to take a pronated grip on the handles."
    },
    {
      "id": 2,
```

```

    "step": "Ensure that your back remains flat, and then remove the weight from the rests, letting the bar hang at arms length. This will be your starting position."
  },
  {
    "id": 3,
    "step": "Perform the movement by flexing the elbows and retracting the shoulder blades, pulling the weight to your chest."
  },
  {
    "id": 4,
    "step": "After a brief pause at the top of the motion, return to the starting position."
  },
  {
    "id": 5,
    "step": "Repeat for the recommended amount of repetitions."
  }
],
"pictures": [
  {
    "start": "http://www.bodybuilding.com/exercises/exerciseImages/sequences/3381/Male/1/3381_1.jpg",
    "finish": "http://www.bodybuilding.com/exercises/exerciseImages/sequences/3381/Male/1/3381_2.jpg"
  }
]
}

```

### c. Non-trivial issues

Exercises demonstrate nested JSON collections. The rest of data is mapped by Foreign – Primary Keys. As demonstrated in the diagram. Most of data depends on data level above and therefore cannot exist independently.



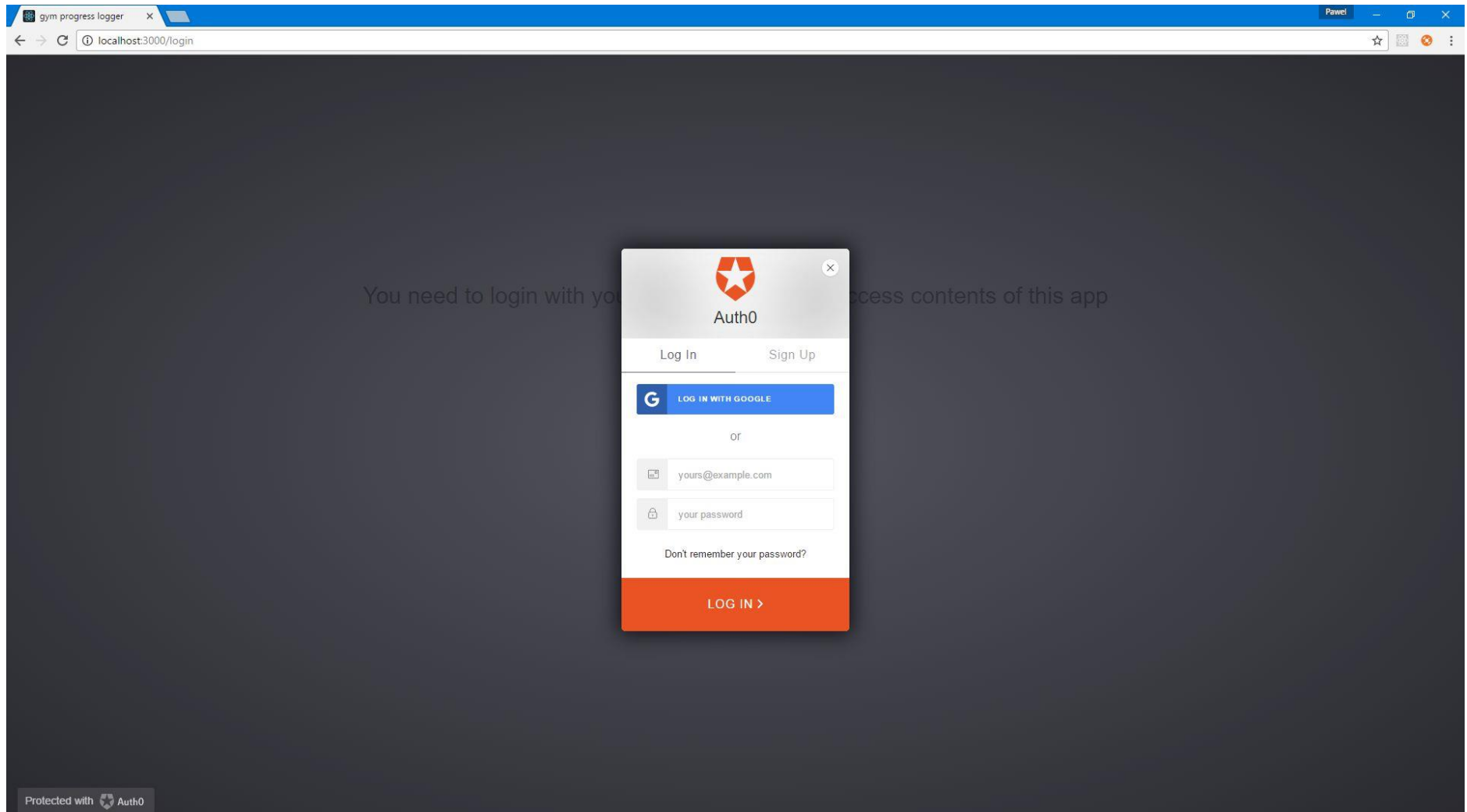
### III. UI Design

#### a. Login (URL: “/login”)



You need to login with your Google account to access contents of this app

Login



Components:

Login

**Stateful**

**Stateless**

**Login**

## b. Home (URL: “/home”)

gym progress logger

Pawel






localhost:3000/home

Welcome Pawel Paszki

Muscles & ExercisesHomeLogout

Go back

SearchSort by: name

	Darell Kent	charts	edit	delete
	Mitch Hitt	charts	edit	delete
	Tyree Layman	charts	edit	delete
	Alphonse Wetzel	charts	edit	delete
	Gus Hathaway	charts	edit	delete

First name

Surname

DOB

Training from

Weight

Height

Add user


Welcome Pawel Paszki

Go back

Search

Sort by: name

Darell	Kent	10/09/1998	12/09/2013	https://s17.postimg.org/b55q322in/5.jpg	111	191	undo	confirm
--------	------	------------	------------	---	-----	-----	------	---------




Mitch Hitt

charts

edit

delete




Tyree Layman

charts

edit

delete




Alphonse Wetzel

charts

edit

delete



Gus Hathaway

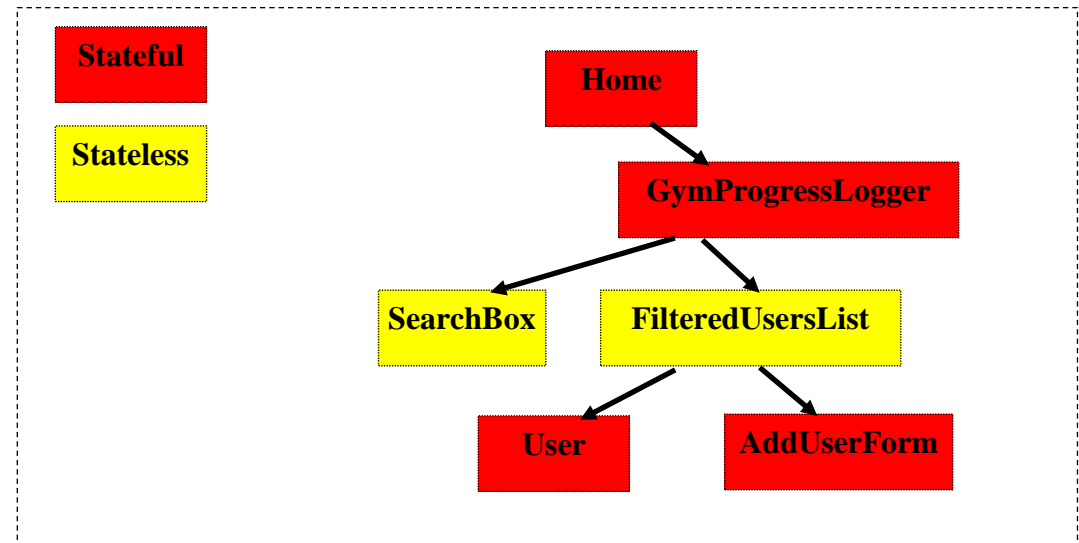
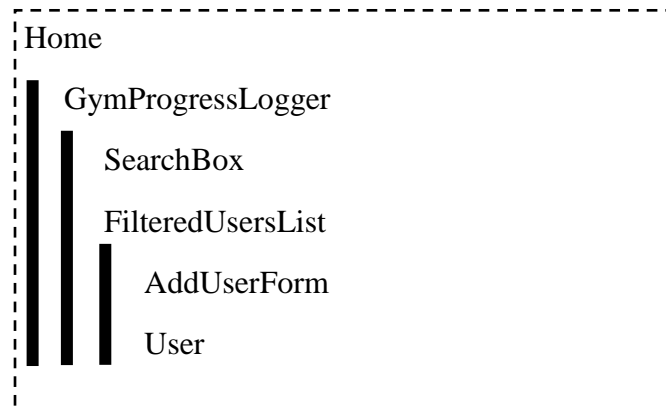
charts

edit

delete

First name	Surname	DOB	Training from	Weight	Height	Add user
------------	---------	-----	---------------	--------	--------	----------

Components:



States:

Home state: { profile: props.auth.getProfile() }

GymProgressLogger state: { search: "", sort: 'dob', users: [], trainingSessions: [], muscleGroupSessions: [], exerciseUnits: [], }

User state: { status: "", initFname: this.props.userItem.first\_name, initSurname: this.props.userItem.surname, initDOB: this.props.userItem.dob, initTrainingFrom: this.props.userItem.training\_from, initWeight: this.props.userItem.body\_weight, initHeight: this.props.userItem.height, picture: this.props.userItem.picture, initPicture: this.props.userItem.picture, id: this.props.userItem.id, first\_name : this.props.userItem.first\_name, surname: this.props.userItem.surname, dob: this.props.userItem.dob, training\_from: this.props.userItem.training\_from, body\_weight: this.props.userItem.body\_weight, height: this.props.userItem.height, }

AddUser state: { first\_name: "", surname: "", dob: "", training\_from: "", body\_weight: "", height: "", }

### c. TrainingSessionList (URL: “/users/:id”)



Darell Kent's sessions

Muscles & Exercises Home

Go back

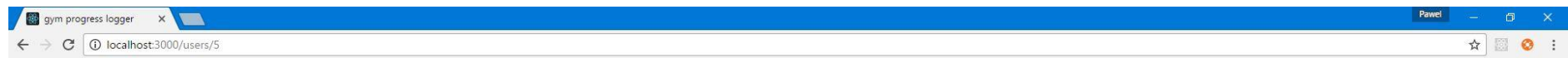
04/07/2016

edit

delete

Enter date

Add session



Darell Kent's sessions

Muscles & Exercises Home

Go back

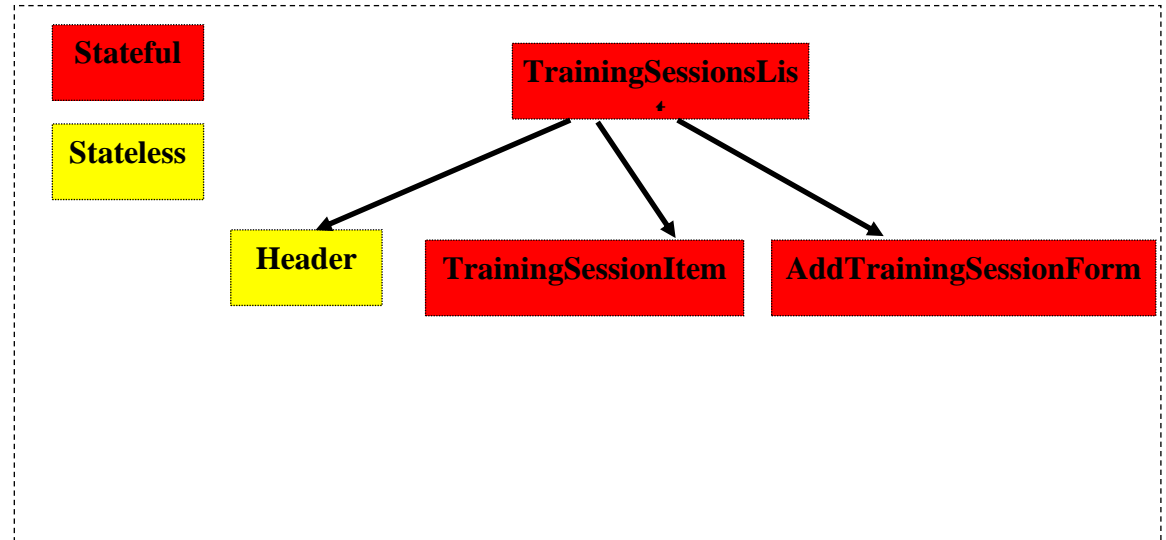
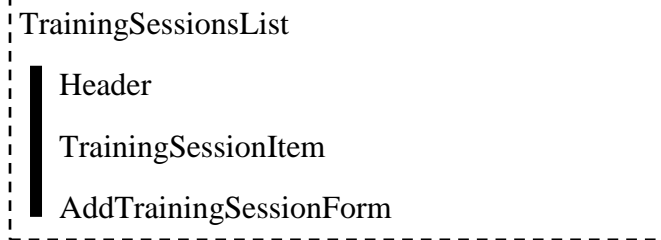
04/07/2016

undo

confirm

Enter date

Add session



States:

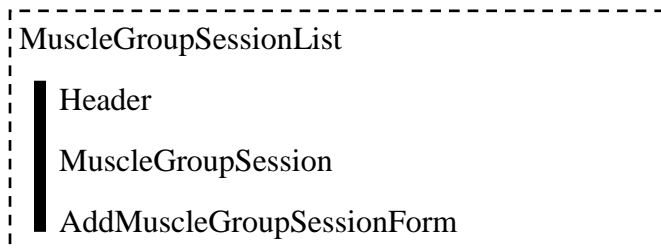
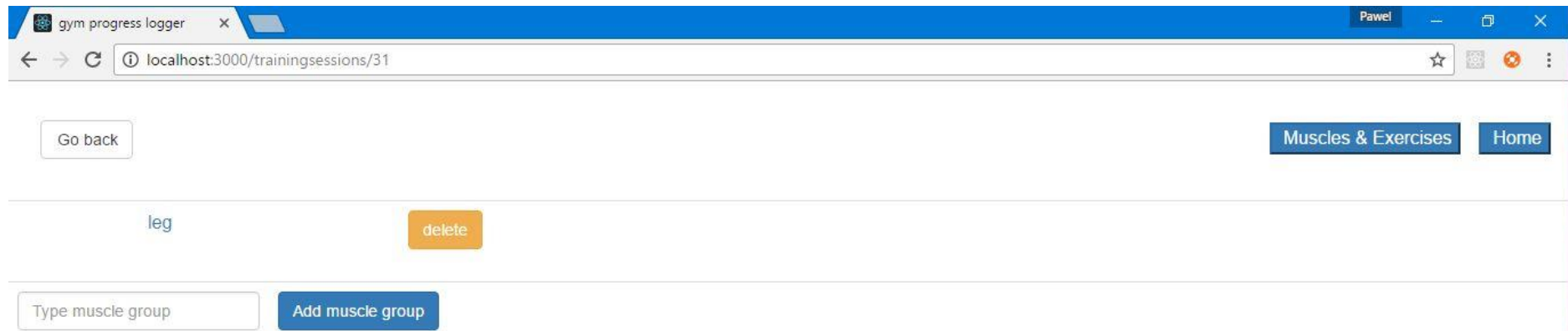
TrainingSessionsList state: {trainingSessions: [], userId: this.props.params.id, users: [], muscleGroupSessions: [], exerciseUnits: [],}

TrainingSessionItem state: {status: "", id : this.props.trainingSessionItem.id, date: this.props.trainingSessionItem.date,  
initDate: this.props.trainingSessionItem.date}

AddTrainingSessionForm state: {status : "", date: ""}



#### d. MuscleGroupSessionList (URL: “/trainingsessions/:id”)



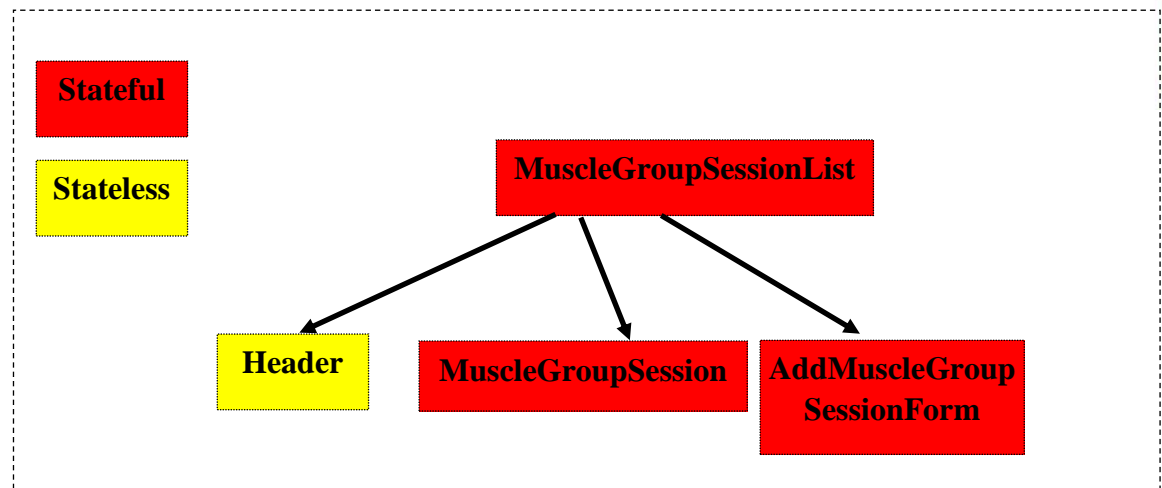
States:

MuscleGroupSessionList state:

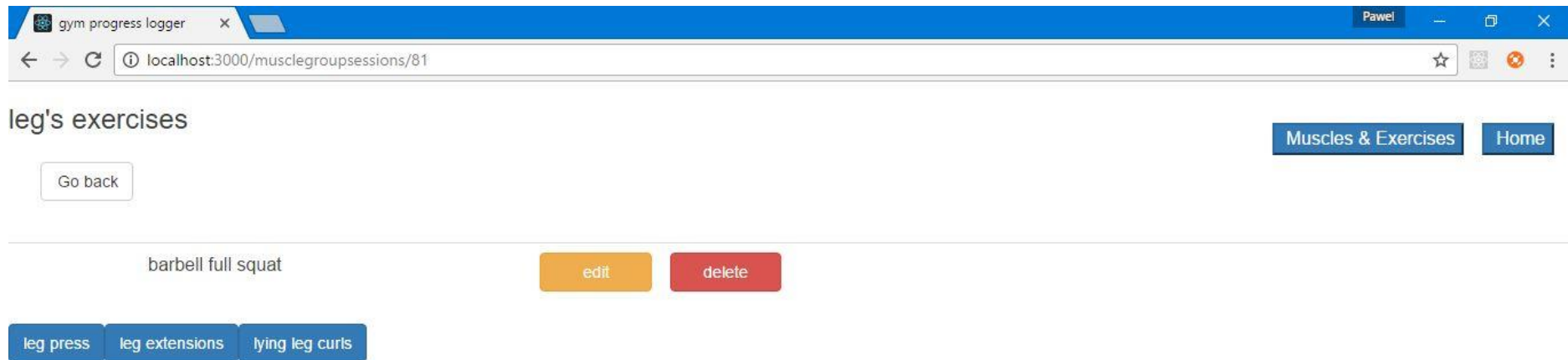
```
{allMuscleGroupSessions: [], allMuscles: [], exerciseUnits: [], trainingSessionId: this.props.params.id, exercises : []}
```

MuscleGroupSession state: {id: this.props.sessionItem.id}

AddMuscleGroupSessionForm state: {status : "", name: ""}



### e. ExerciseUnitList (URL: “/musclegroupsessions/:id”)



ExerciseUnitList

Header

Exercise Unit

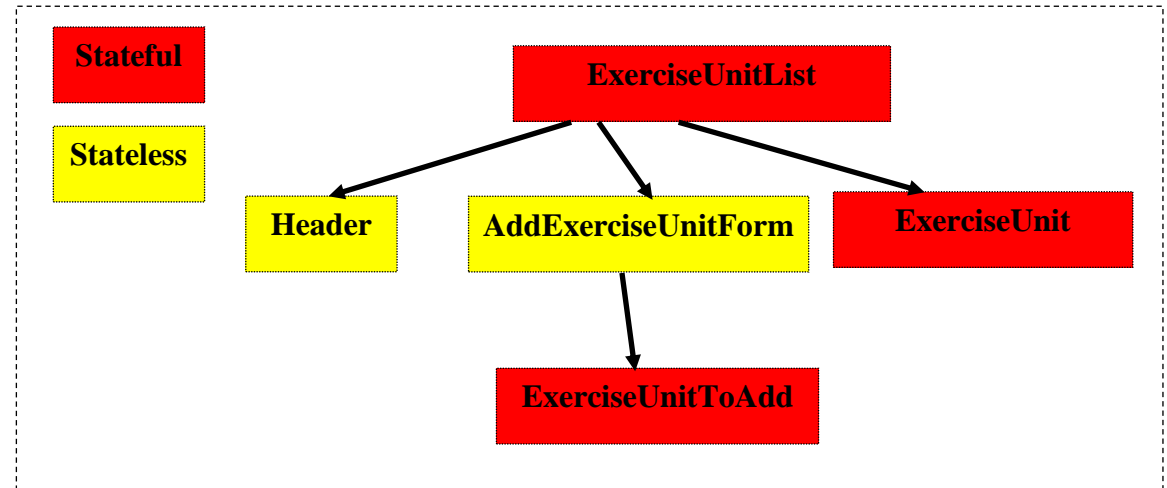
AddExerciseUnitForm

ExerciseUnitToAdd

States:

ExerciseUnitList state: {allExerciseUnits: [],

allExercises : [], muscleGroup: ", muscleGroupSessionId: this.props.params.id}



```
ExerciseUnit state: {status: ", id: this.props.exerciseUnit.id, name: this.props.exerciseUnit.name, weight: this.props.exerciseUnit.weight,  
    number_of_series: this.props.exerciseUnit.number_of_series, number_of_reps: this.props.exerciseUnit.number_of_reps}  
ExerciseUnitToAdd state: {status : ", name: this.props.exerciseUnitToAdd.name}
```

## f. ChartGenerator (URL: “/charts/:id”)

gym progress logger

Pawel

localhost:3000/charts/1

☆

Go back

Muscles & Exercises

Home

Type an exercise name

Date from: dd/mm/yyyy

Date to: dd/mm/yyyy

Generate chart

Weight

Date

Go back

Muscles & Exercises

Home

barbell flat

Date from: dd/mm/yyyy

Date to: dd/mm/yyyy

Generate chart

- barbell flat bench press

Weight

Date

Go back

Muscles & Exercises

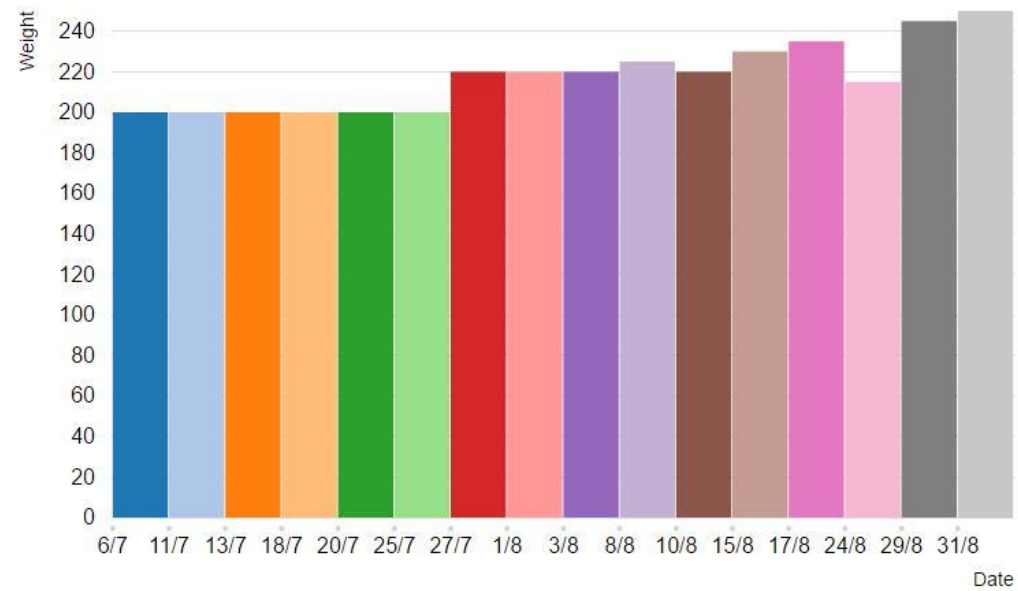
Home

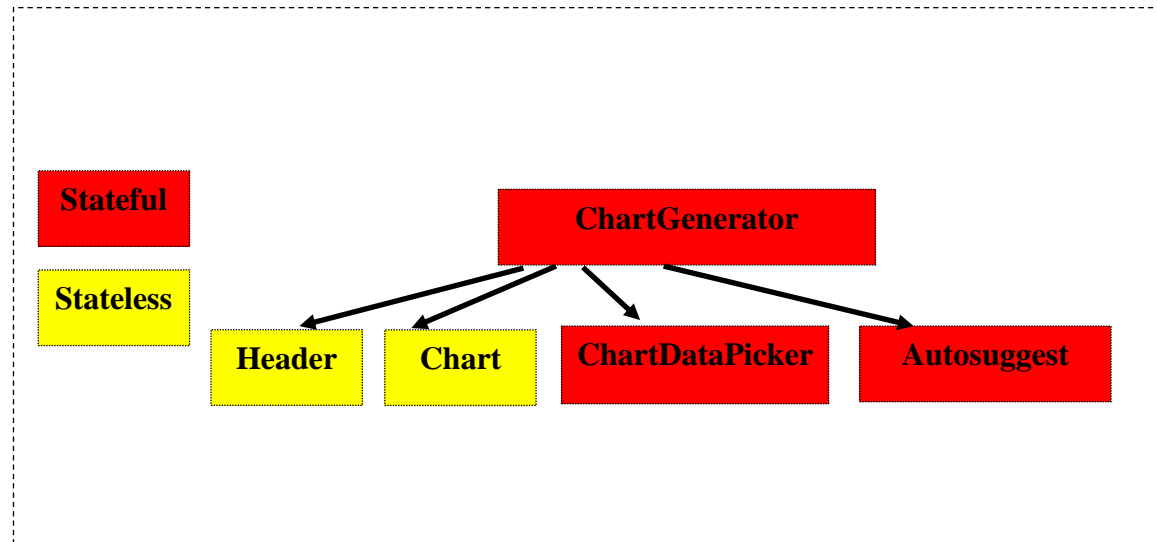
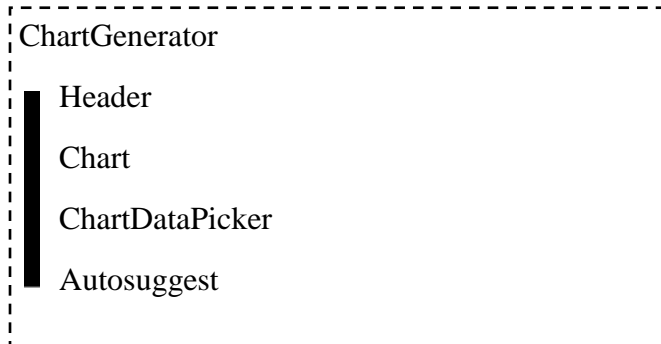
standing calf raise

11/11/2015

11/11/2016

Generate chart





States:

ChartGenerator state: { value: "", data: [], users : [], suggestions: [], exerciseUnits : [], muscleGroupSessions : [], trainingSessions : [],  
exercises : [], }

ChartDataPicker state: { status : "", date\_from: "", date\_to: "", user: "", }

Autosuggest is a 3<sup>rd</sup> party component – I was not changing anything in its state

g. MuscleList (URL: “/muscles”)

The screenshot shows a web browser window with the title 'gym progress logger'. The address bar displays 'localhost:3000/muscles'. The page content is titled 'Muscles' and includes a 'Go back' button. On the right side, there are two navigation links: 'Muscles & Exercises' and 'Home'. The main content area is a table with seven rows, each representing a muscle. Each row contains the muscle name and two buttons: 'edit' (blue) and 'delete' (orange). At the bottom of the page, there is a text input field with the placeholder 'Add new muscle' and an 'Add muscle' button (blue).

Muscle	edit	delete
chest		
biceps		
calf		
triceps		
leg		
shoulders		
back		

Add new muscle



Muscles

Muscles & ExercisesHome

Go back

chest

undo

confirm

biceps

edit

delete

calf

edit

delete

triceps

edit

delete

leg

edit

delete

shoulders

edit

delete

back

edit

delete

Add new muscle

Add muscle

MuscleList

Header

AddMuscleForm

Muscle

Stateful

Stateless

MuscleList

Header

AddMuscleForm

Muscle

States:

MuscleList state: {allMuscles: [], exercises: [],}

AddMuscleForm state: {name: ""}

Muscle state: {status: "", id: this.props.muscle.id, name: this.props.muscle.name, initName: this.props.muscle.name}

## h. ExerciseList (URL: “/musclegroupexercises/:id”)

gym progress logger x

Paweł

localhost:3000/musclegroupexercises/1

### Exercises (chest)

Muscles & Exercises Home

Go back

barbell flat bench press	edit	delete
barbell incline bench press	edit	delete
barbell decline bench press	edit	delete

Add exercise

## Exercises (chest)

Muscles & Exercises

Home

Go back

barbell flat bench press	<div>edit</div> <div>delete</div>
barbell incline bench press	<div>edit</div> <div>delete</div>
barbell decline bench press	<div>edit</div> <div>delete</div>

Exercise name

Step 1

Step 2

Step 3

Step 4

Step 5

Picture 1 url

Picture 2 url

Undo

Confirm

Exercises (chest)

Go back

barbell flat bench press

undo

confirm

barbell incline bench press

edit

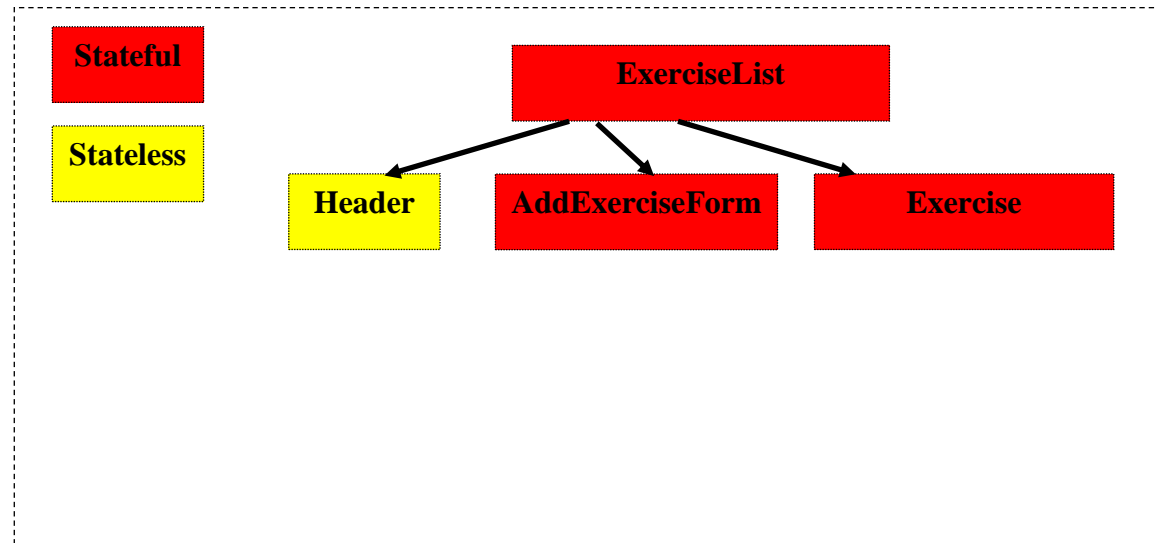
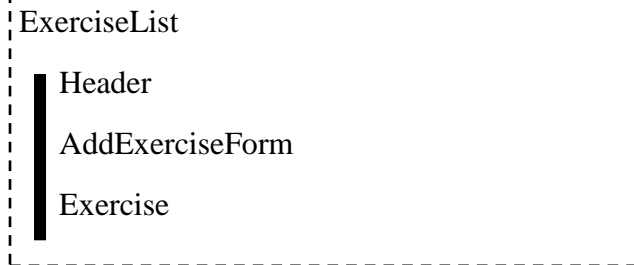
delete

barbell decline bench press

edit

delete

Add exercise



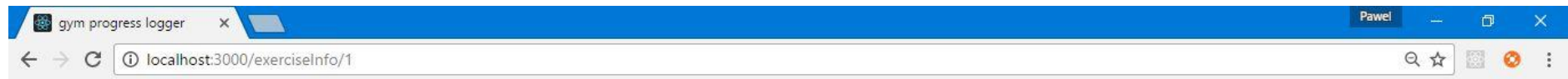
States:

ExerciseList state: {exercises: [], muscles: [], muscleId: this.props.params.id, group: ","}

AddExerciseForm state: {status: " , name: " , step1: " , step2: " , step3: " , step4: " , step5: " , picture1: " , picture2: " , group: this.props.muscleGroup}

Exercise state: {status: " , id: this.props.exercise.id, name: this.props.exercise.name, group: this.props.exercise.group, initName: this.props.exercise.name, descriptions: this.props.exercise.descriptions, pictures: this.props.exercise.pictures,}

## i. ExerciseInfo (URL: “exerciseinfo/:id”)



### barbell flat bench press

[Muscles & Exercises](#) [Home](#)

[Go back](#)

Lie back on a flat bench. Lift the bar from the rack and hold it straight over you with your arms locked. This will be your starting position.

From the starting position, breathe in and begin coming down slowly until the bar touches your middle chest.

After a brief pause, push the bar back to the starting position as you breathe out. Focus on pushing the bar using your chest muscles. Lock your arms and squeeze your chest in the contracted position at the top of the motion, hold for a second and then start coming down slowly again. Tip: Ideally, lowering the weight should take about twice as long as raising it.

Repeat the movement for the prescribed amount of repetitions.

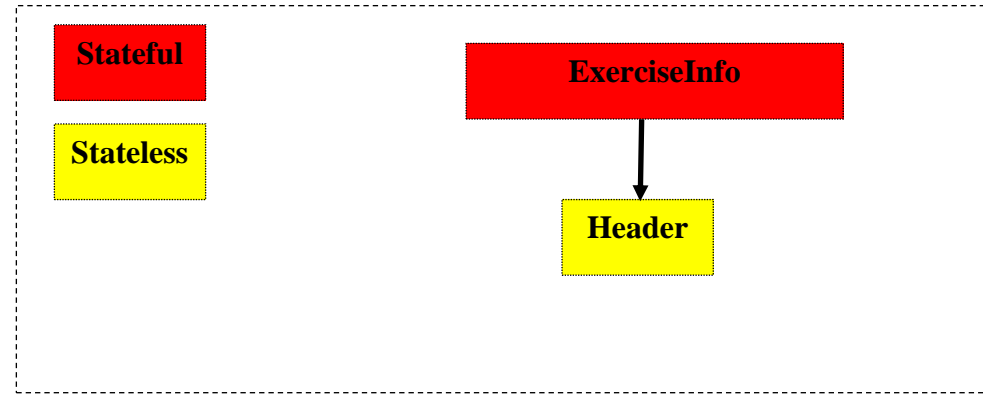
When you are done, place the bar back in the rack.





States:

ExerciseList state: {exercise: "", id: this.props.params.id, path: "",}





## IV. Extra Features

- Authorization (including sign up and login) is provided by Auth0 (<https://auth0.com/>). Starter code was downloaded and merged with my existing project.
- Autosuggest component was used to help picking exercise name to generate the graph (available here: <https://github.com/moroshko/react-autosuggest>). .ENV file is required to use Auth0. I have included mine with my credentials and I kindly request to remove this file after my application is graded.
- React Easy Chart (Bar Chart) component was used to generate graphs (available here: <https://www.npmjs.com/package/react-easy-chart>)
- CRUD supported by mock json-server and JQuery to handle with responses

## V. Independent Learning

- Generating meaningful data for graphs and other data processing throughout the app required doing research about javascript syntax, like for loops, indexOf and other methods and utils (I needed to parse my date strings to create Date objects and compare them to check, if given dates are in certain range for chart generation). For some other data processing methods I needed to find the way to extract data the way I want, so I found .pluck method in lodash library, which was very useful in extracting specified attribute (ie exercisesAdded = `_.pluck(this.props.exerciseUnits, 'name')`) and creating an array of the specified attribute values.
- All forms have basic input validation, ie date format validation (available here: <http://stackoverflow.com/questions/6177975/how-to-validate-date-with-format-mm-dd-yyyy-in-javascript>) or checking whether number or string is where its expected
- JQuery for me can be considered as independent learning as well.
- Bootstrap documentation had to be looked into to structure the code in desired way and to use different types of styling, ie button colors, table types, etc (some inline css styling had to be used to overcome limited Bootstrap options)