Dokumentation

Aufgabe 2
Anwendungsprojekt Webtechnologien
BITI 18
Weltler Martin

Version 001

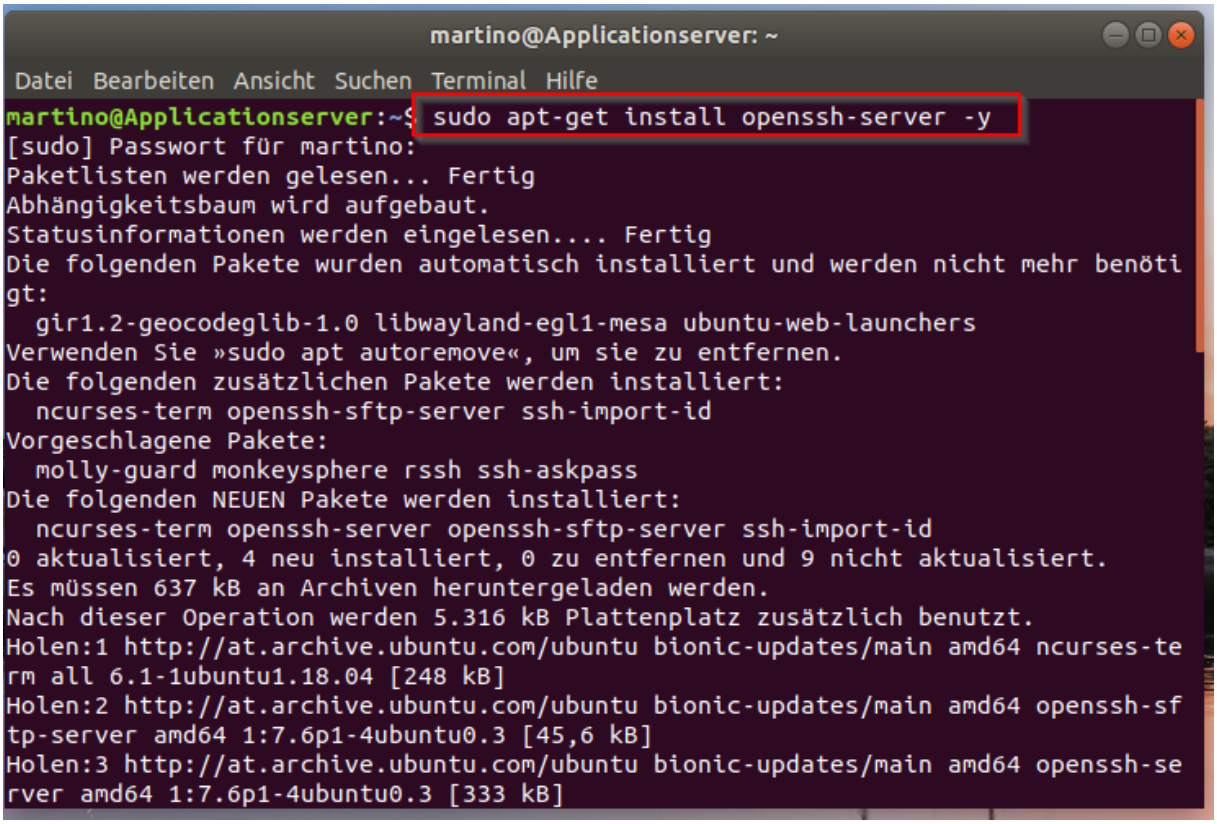
Inhaltsverzeichnis

1 Meteor

Plattform: Ubuntu VM (Virtual Box)

SSH-Zugang für die virtuelle Maschine erlauben:

`sudo apt-get install openssh-server -y`



Mittels putty mit der virtuellen Maschine veebinden:

curl installieren:

`sudo apt install curl`

```
martino@Applicationserver:~$ sudo apt install curl
[sudo] Passwort für martino:
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
Die folgenden Pakete wurden automatisch installiert und werden nicht mehr benötigt:
  girl.2-geocodeglib-1.0 libwayland-egl1-mesa ubuntu-web-launchers
Verwenden Sie »sudo apt autoremove«, um sie zu entfernen.
Die folgenden zusätzlichen Pakete werden installiert:
  libcurl4
Die folgenden NEUEN Pakete werden installiert:
```

neuestes Meteor Release installieren:

curl https://install.meteor.com/ | sh

```
martino@Applicationserver:~$ curl https://install.meteor.com/ | sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  7915    0  7915    0     0  12664      0 --:--:-- --:--:-- --:--:-- 12664
Downloading Meteor distribution
#############                                                       18,5%
```

2 Todo-Applikation

2.1 Todo App installieren:

Verzeichnis anlegen:

mkdir <Name>

cd <Name>

Repository herunterladen:

git clone https://github.com/meteor/todos.git

cd todos

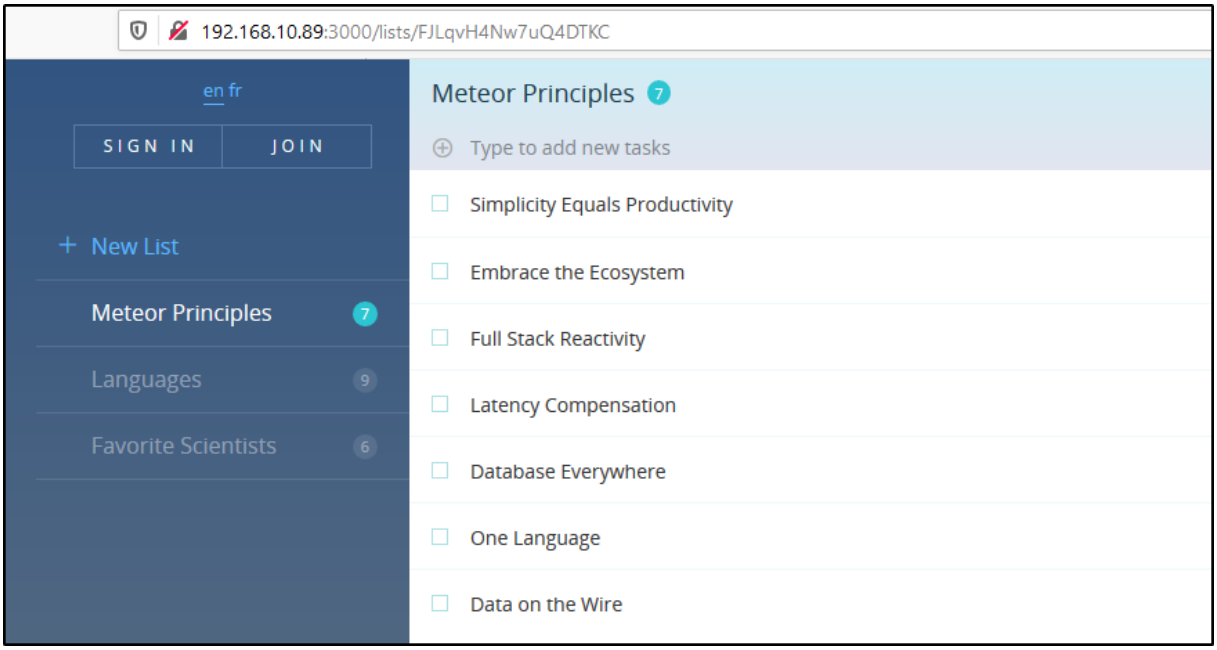
App starten:

meteor npm install

meteor

```
martino@meteor:~$ mkdir tutorial
martino@meteor:~$ cd tutorial/
martino@meteor:~/tutorial$ git clone https://github.com/meteor/todos.git
Cloning into 'todos'...
remote: Enumerating objects: 36, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 4814 (delta 9), reused 19 (delta 4), pack-reused 4778
Receiving objects: 100% (4814/4814), 1.19 MiB | 2.10 MiB/s, done.
Resolving deltas: 100% (2904/2904), done.
martino@meteor:~/tutorial$ cd todos/
martino@meteor:~/tutorial/todos$ meteor npm install

> bcrypt@5.0.0 install /home/martino/tutorial/todos/node_modules/bcrypt
> node-pre-gyp install --fallback-to-build

node-pre-gyp WARN Using request for node-pre-gyp https download
[bcrypt] Success: "/home/martino/tutorial/todos/node_modules/bcrypt/lib/binding/napi-v3/
bcrypt_lib.node" is installed via remote

> core-js@2.6.11 postinstall /home/martino/tutorial/todos/node_modules/core-js
> node -e "try{require('./postinstall')}catch(e){}"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilling Java
Script standard library!

The project needs your help! Please consider supporting of core-js on Open Collective or
 Patreon:
> https://opencollective.com/core-js
> https://www.patreon.com/zloirock

Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job -)


> electron@2.0.18 postinstall /home/martino/tutorial/todos/node_modules/electron
> node install.js

npm WARN todos No repository field.
npm WARN todos No license field.

added 468 packages from 590 contributors and audited 458 packages in 15.945s

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

martino@meteor:~/tutorial/todos$ meteor
[[[[[ ~/tutorial/todos ]]]]]

=> Started proxy.
=> Started MongoDB.
=> Started your app.

=> App running at: http://localhost:3000/
```

Kontrolle:


192.168.10.89:3000/lists/FJLqvH4Nw7uQ4DTKC
en fr
SIGN IN
JOIN
New List
Meteor Principles 7
Languages 9
Favorite Scientists 6
Meteor Principles 7
Type to add new tasks
Simplicity Equals Productivity
Embrace the Ecosystem
Full Stack Reactivity
Latency Compensation
Database Everywhere
One Language
Data on the Wire

3 Individualisieren

Individualisieren geht am besten, wenn man die App komplett selbst erstellt, da man sonst die Zusammenhänge der Files kaum oder nur sehr schwer versteht.

Dazu eignet sich das „simple-todos" Repository von Meteor, da hier nur die notwendigsten Dateien vorhanden sind.

App erstellen:

`meteor create simple-todos`

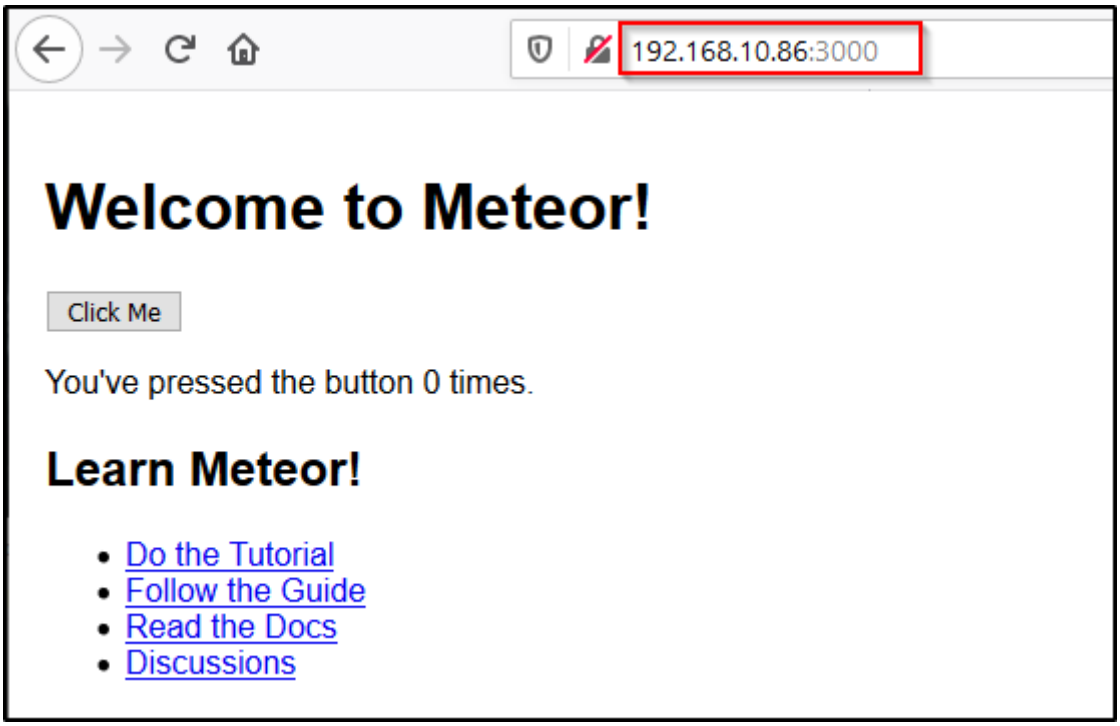
ins Verzeichnis wechseln und App starten:

`cd simple-todos`

`meteor`

```
martino@Applicationserver:~$ meteor create simple-todos
Created a new Meteor app in 'simple-todos'.

To run your new app:
  cd simple-todos
  meteor

If you are new to Meteor, try some of the learning resources here:
  https://www.meteor.com/tutorials

When you're ready to deploy and host your new Meteor application, check out Galaxy:
  https://www.meteor.com/hosting

To start with a different app template, try one of the following:

  meteor create --bare       # to create an empty app
  meteor create --minimal    # to create an app with as few Meteor packages as possible
  meteor create --full       # to create a more complete scaffolded app
  meteor create --react      # to create a basic React-based app
  meteor create --typescript # to create an app using TypeScript and React

martino@Applicationserver:~$ cd simple-todos
martino@Applicationserver:~/simple-todos$ meteor
[[[[[ ~/simple-todos ]]]]]

=> Started proxy.
=> Started MongoDB.
=> Started your app.

=> App running at: http://localhost:3000/
```

Seite besuchen und kontrollieren ob alle funktioniert hat:


192.168.10.86:3000
Welcome to Meteor!
Click Me
You've pressed the button 0 times.
Learn Meteor!
Do the Tutorial
Follow the Guide
Read the Docs
Discussions

3.1 Todo App erstellen:

sudo nano client/main.html

```
  GNU nano 2.9.3                                main.html

<head>
  <title>simple-todos</title>
</head>

<body>
  <h1>Welcome to Meteor!</h1>

  {{> hello}}
  {{> info}}
</body>

<template name="hello">
  <button>Click Me</button>
  <p>You've pressed the button {{counter}} times.</p>
</template>

<template name="info">
  <h2>Learn Meteor!</h2>
  <ul>
    <li><a href="https://www.meteor.com/try" target="_blank">Do the Tutorial</a></li>
    <li><a href="http://guide.meteor.com" target="_blank">Follow the Guide</a></li>
    <li><a href="https://docs.meteor.com" target="_blank">Read the Docs</a></li>
    <li><a href="https://forums.meteor.com" target="_blank">Discussions</a></li>
  </ul>
</template>
```

alles außer den head löschen:

```
  GNU nano 2.9.3

head>
  <title>simple-todos</title>
</head>
```

Verzeichnisse anlegen:

mkdir imports

cd imports/

mkdir ui && mkdir api

Eine body.html erstellen:

sudo nano ui/body.html

```
  GNU nano 4.8                                                ui/body.html
<body>
    <div class="container">
      <header>
        <h1>Todo List</h1>
      </header>
      <ul>
        {{#each tasks}}
          {{> task}}
        {{/each}}
      </ul>
    </div>
  </body>
  <template name="task">
    <li>{{text}}</li>
  </template>
```

```
<body>
    <div class="container">
      <header>
        <h1>Todo List</h1>
      </header>

      <ul>
        {{#each tasks}}
          {{> task}}
        {{/each}}
      </ul>
    </div>
  </body>

  <template name="task">
    <li>{{text}}</li>
  </template>
```

Eine body.js erstellen:

```
sudo nano ui/body.js
```

```
  GNU nano 4.8                                                 ui/body.js
import { Template } from 'meteor/templating';
import './body.html';

Template.body.helpers({
  tasks: [
    { text: 'This is task 1' },
    { text: 'This is task 2' },
    { text: 'This is task 3' },
  ],
});
```

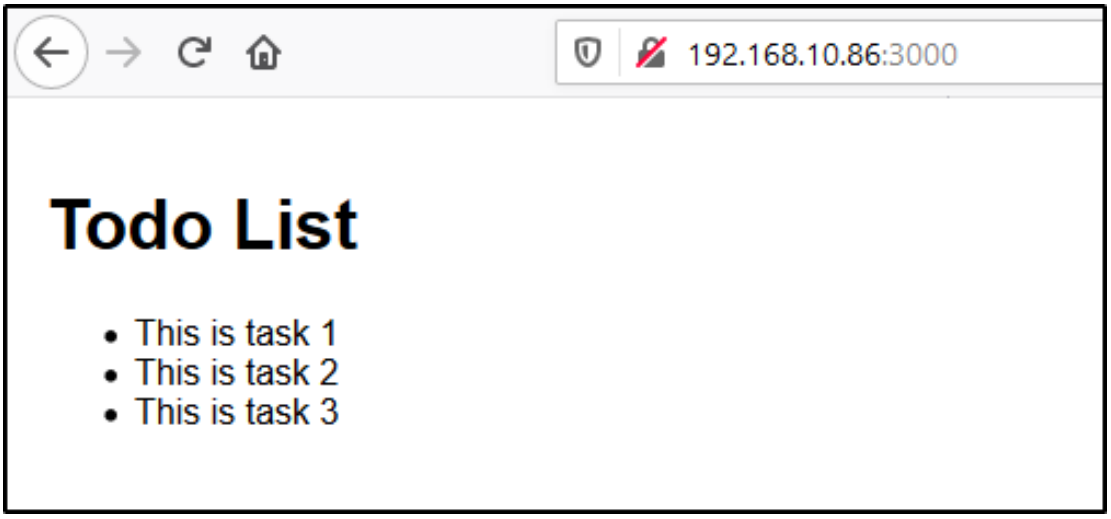
```
import { Template } from 'meteor/templating';

import './body.html';

Template.body.helpers({
  tasks: [
    { text: 'This is task 1' },
    { text: 'This is task 2' },
    { text: 'This is task 3' },
  ],
});
```

Danach main.js bearbeiten und auf das eben erstellte body.js „verweisen“:

```
sudo nano ../client/main.js
```

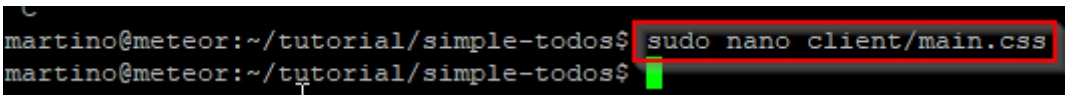

```
import '../imports/ui/body.js';
```

Danach die App wieder starten und kontrollieren:



Eigenes Styling für main.css:

```
sudo nano client/main.css
```



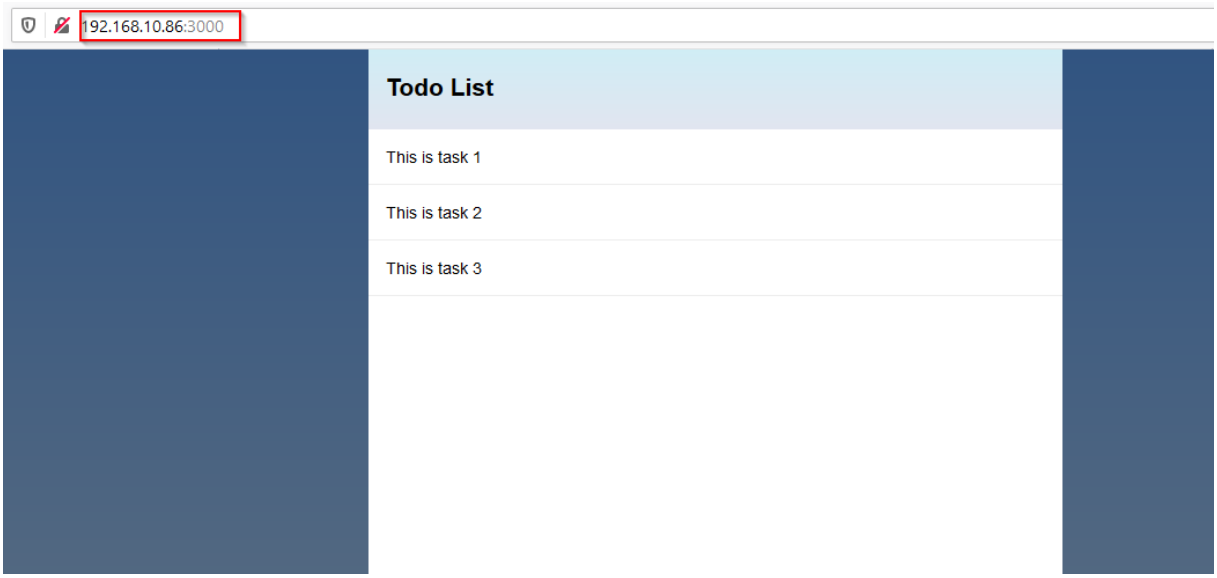
Code einfügen:

```
/* CSS declarations go here */
body {
  font-family: sans-serif;
  background-color: #315481;
  background-image: linear-gradient(to bottom, #315481, #918e82 100%);
  background-attachment: fixed;

  position: absolute;
  top: 0;
  bottom: 0;
  left: 0;
  right: 0;

  padding: 0;
  margin: 0;

  font-size: 14px;
}

.container {
  max-width: 600px;
  margin: 0 auto;
  min-height: 100%;
  background: white;
}

header {
  background: #d2edf4;
  background-image: linear-gradient(to bottom, #d0edf5, #e1e5f0 100%);
  padding: 20px 15px 15px 15px;
  position: relative;
}

#login-buttons {
  display: block;
}

h1 {
  font-size: 1.5em;
  margin: 0;
  margin-bottom: 10px;
  display: inline-block;
  margin-right: 1em;
}

form {
  margin-top: 10px;
  margin-bottom: -10px;
  position: relative;
}

.new-task input {
  box-sizing: border-box;
  padding: 10px 0;
  background: transparent;
  border: none;
  width: 100%;
  padding-right: 80px;
  font-size: 1em;
}
```

```
.new-task input:focus{
  outline: 0;
}

ul {
  margin: 0;
  padding: 0;
  background: white;
}

.delete {
  float: right;
  font-weight: bold;
  background: none;
  font-size: 1em;
  border: none;
  position: relative;
}

li {
  position: relative;
  list-style: none;
  padding: 15px;
  border-bottom: #eee solid 1px;
}

li .text {
  margin-left: 10px;
}

li.checked {
  color: #888;
}

li.checked .text {
  text-decoration: line-through;
}

li.private {
  background: #eee;
  border-color: #ddd;
}

header .hide-completed {
  float: right;
}

.toggle-private {
  margin-left: 5px;
}

@media (max-width: 600px) {
  li {
    padding: 12px 15px;
  }

  .search {
    width: 150px;
    clear: both;
  }
```

```
  .new-task input {
    padding-bottom: 5px;
  }
}
```

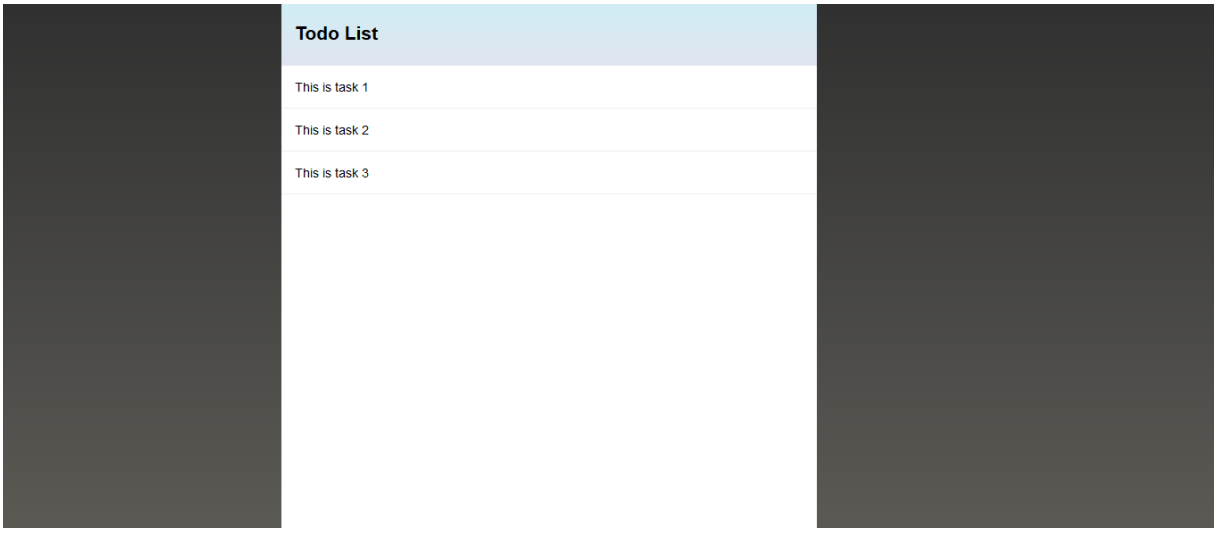
Danach sollte die Seite die App folgendermaßen aussehen:



Danach kann man das Design nach Belieben verändern. Für einen dunkleren Hintergrund im body background-image bearbeiten:

z.B.: background-image:

```
background-image: linear-gradient(to bottom, #303030, #918e82 100%);
```



Um Aufgaben (tasks) speichern zu können, muss man für Meteor collections verwenden.

Collection erstellen:

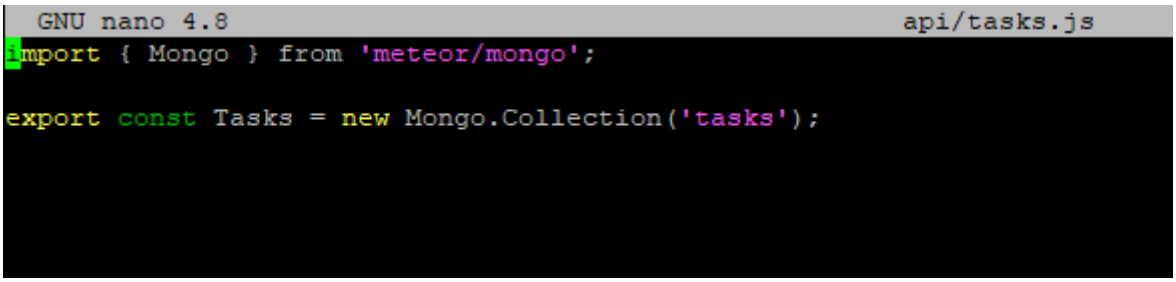
in dem Verzeichnis imports ein Verzeichnis api erstellen und darin tasks.js erzeugen

```
cd imports
```

sudo nano api/tasks.js

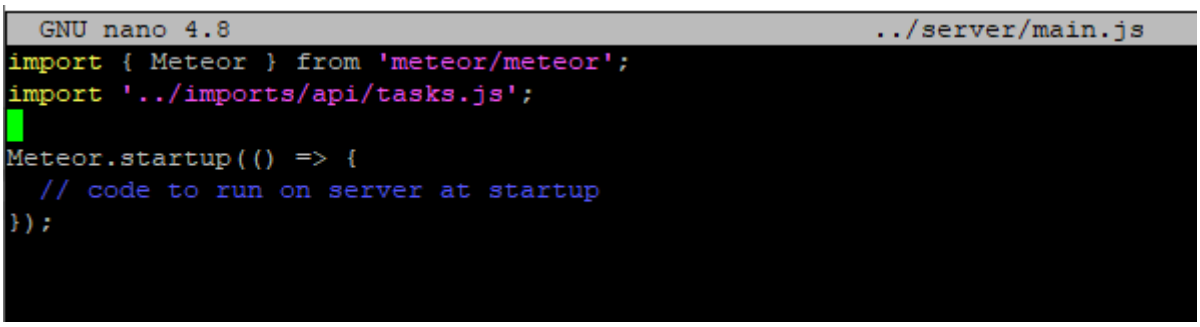
Danach folgenden Zeilen in tasks.js einfügen:

```
import { Mongo } from 'meteor/mongo';

export const Tasks = new Mongo.Collection('tasks');
```



Danach ins server Verzeichnis wechseln und in main.js die eben erstellte tasks.js mitimportieren lassen:

```
import '../imports/api/tasks.js';
```



Durch das Importieren von task.js auf den Server wird die MongoDB Collection erstellt und eingerichtet, um die Daten an den Client zu übertragen.

Jetzt muss noch der Clientseitige Code geändert werden, um die „tasks" aus der Collection abzurufen:

Im imports Verzeichnis body.js öffnen und folgende Zeilen einfügen:

```
import { Tasks } from '../api/tasks.js';

Template.body.helpers({
    tasks() {
        return Tasks.find({});
      },
```

```
  GNU nano 4.8                                                  ui/body.js
import { Template } from 'meteor/templating';
import { Tasks } from '../api/tasks.js';
import './body.html';

Template.body.helpers({
    tasks() {
        return Tasks.find({});
    },
});
```

Zur Kontrolle kann man serverseitig bereits einen Task zur collection hinzufügen.

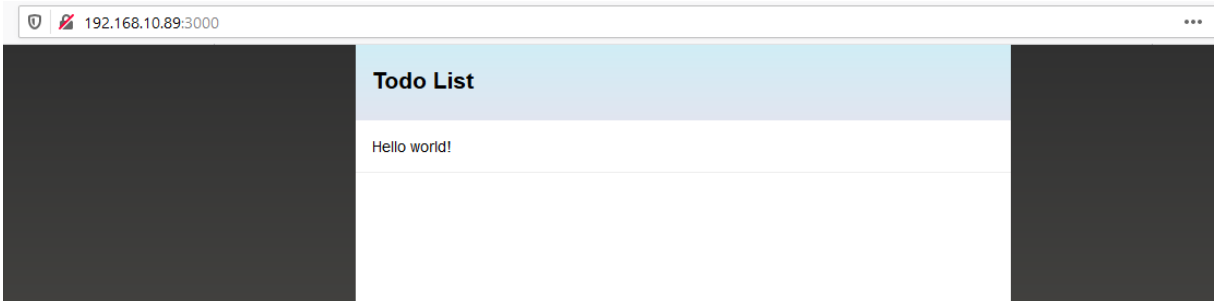
Dazu während die App läuft mit einer zweiten shell ins Projektverzeichnis wechseln und folgende Befehle ausführen:

`meteor mongo`

`db.tasks.insert({ text: "Hello world!", createdAt: new Date() });`

```
martino@meteor:~/tutorial/simple-todos$ meteor mongo
MongoDB shell version v4.2.5
connecting to: mongodb://127.0.0.1:3001/meteor?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("a588ef9a-d9b4-41a3-b136-082b42362b06") }
MongoDB server version: 4.2.5
Server has startup warnings:
2020-06-15T17:59:20.919+0000 I  STORAGE  [initandlisten]
2020-06-15T17:59:20.919+0000 I  STORAGE  [initandlisten] ** WARNING: Using the XFS filesystem is strongly
h the WiredTiger storage engine
2020-06-15T17:59:20.919+0000 I  STORAGE  [initandlisten] **          See http://dochub.mongodb.org/core/p
tem
2020-06-15T17:59:22.332+0000 I  CONTROL  [initandlisten]
2020-06-15T17:59:22.332+0000 I  CONTROL  [initandlisten] ** WARNING: soft rlimits too low. rlimits set to
, 1048576 files. Number of processes should be at least 524288 : 0.5 times number of files.
meteor:PRIMARY> db.tasks.insert({ text: "Hello world!", createdAt: new Date() });
WriteResult({ "nInserted" : 1 })
meteor:PRIMARY>
```

Nun sollte die App folgendermaßen aussehen:



Um ein task (Aufgabe) hinzufügen zu können, wird eine „Form" (Formular) erstellt.

Dazu im ui Verzeichnis body.html folgendermaßen anpassen:

`cd imports/ui/`

`sudo nano body.html`

und folgende Zeilen im header unterhalb von h1 einfügen:

```
<form class="new-task">
```

```
    <input type="text" name="text" placeholder="Type to add new tasks" />
  </form>
```

```
 GNU nano 2.9.3                                  body.html

<body>
    <div class="container">
      <header>
        <h1>Todo List</h1>

        <form class="new-task">
            <input type="text" name="text" placeholder="Type to add new tasks" />
          </form>
      </header>


      <ul>
        {{#each tasks}}
          {{> task}}
        {{/each}}
      </ul>
    </div>
```

Danach body.js bearbeiten, damit der eingegebene Text in diesen „form" auch gespeichert wird:

```
sudo nano body.js
```

Folgenden code unterhalb einfügen:

```
Template.body.events({
    'submit .new-task'(event) {
      // Prevent default browser form
submit
      event.preventDefault();

      // Get value from form element
      const target = event.target;
      const text = target.text.value;

      // Insert a task into the
collection
      Tasks.insert({
        text,
        createdAt: new Date(), //
current time
      });

      // Clear form
      target.text.value = '';
    },
  });
```

```
 GNU nano 4.8
import { Template } from 'meteor/templating';
import { Tasks } from '../api/tasks.js';
import './body.html';

Template.body.helpers({
    tasks() {
        return Tasks.find({});
      },
});

Template.body.events({
    'submit .new-task'(event) {
      // Prevent default browser form submit
      event.preventDefault();

      // Get value from form element
      const target = event.target;
      const text = target.text.value;

      // Insert a task into the collection
      Tasks.insert({
        text,
        createdAt: new Date(), // current time
      });

      // Clear form
      target.text.value = '';
    },
  });
```

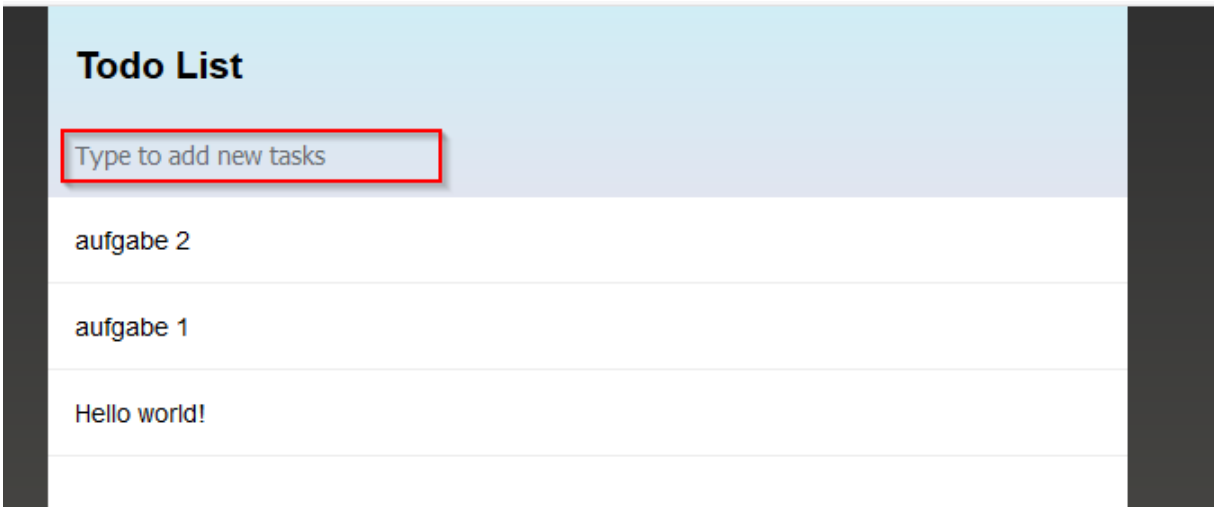
Wenn man die Reihenfolge wie die Tasks angezeigt werden ändern möchte muss man folgenden code in body.js einfügen:

```
return Tasks.find({}, { sort: { createdAt: -1 } });
```

```
 GNU nano 2.9.3                                          body.js

import { Template } from 'meteor/templating';
import { Tasks } from '../api/tasks.js';
import './body.html';

Template.body.helpers({
    tasks() {
        return Tasks.find({}, { sort: { createdAt: -1 } });
      },
});

Template.body.events({
    'submit .new-task'(event) {
      // Prevent default browser form submit
      event.preventDefault();
```

Danach werden die neuesten Tasks oberhalb angezeigt.

Nun kann man Tasks in das Textfeld eingeben und mit betätigen der Enter-Taste speichern:



Tasks bearbeiten und entfernen:

Buttons zu den Task-Elementen hinzufügen:

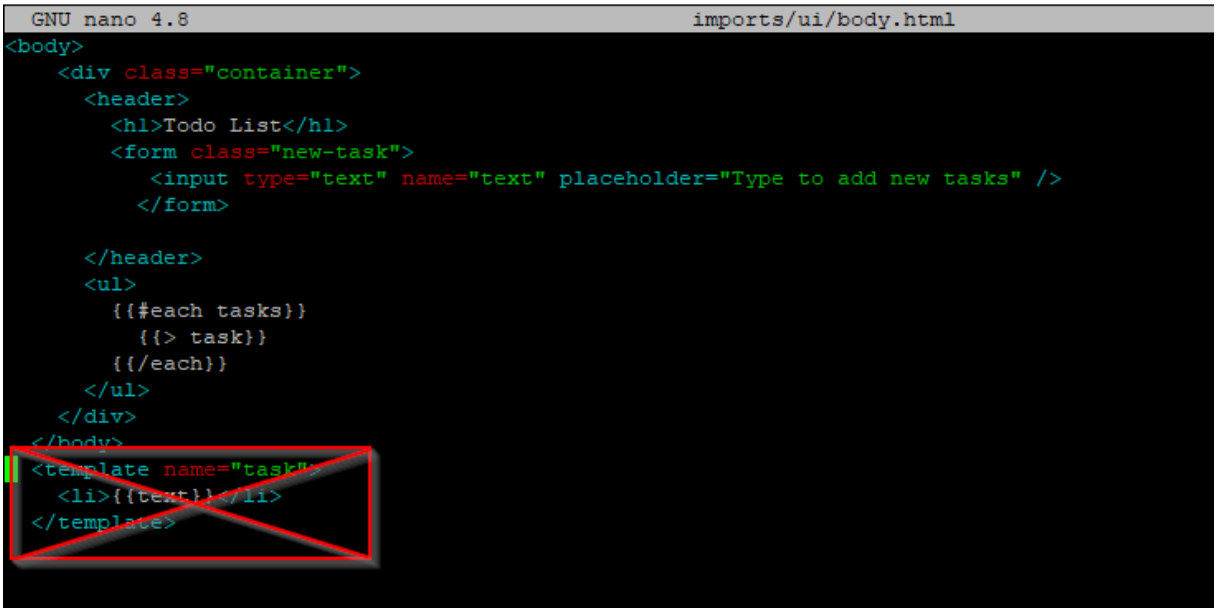
sudo nano imports/ui/task.html

```
 GNU nano 4.8                                    imports/ui/task.html
<template name="task">
    <li class="{{#if checked}}checked{{/if}}">
      <button class="delete">&times;</button>
      <input type="checkbox" checked="{{checked}}" class="toggle-checked" />
      <span class="text">{{text}}</span>
    </li>
  </template>
```

```
<template name="task">
    <li class="{{#if checked}}checked{{/if}}">
      <button class="delete">&times;</button>

      <input type="checkbox" checked="{{checked}}" class="toggle-checked"
/>

      <span class="text">{{text}}</span>
    </li>
  </template>
```

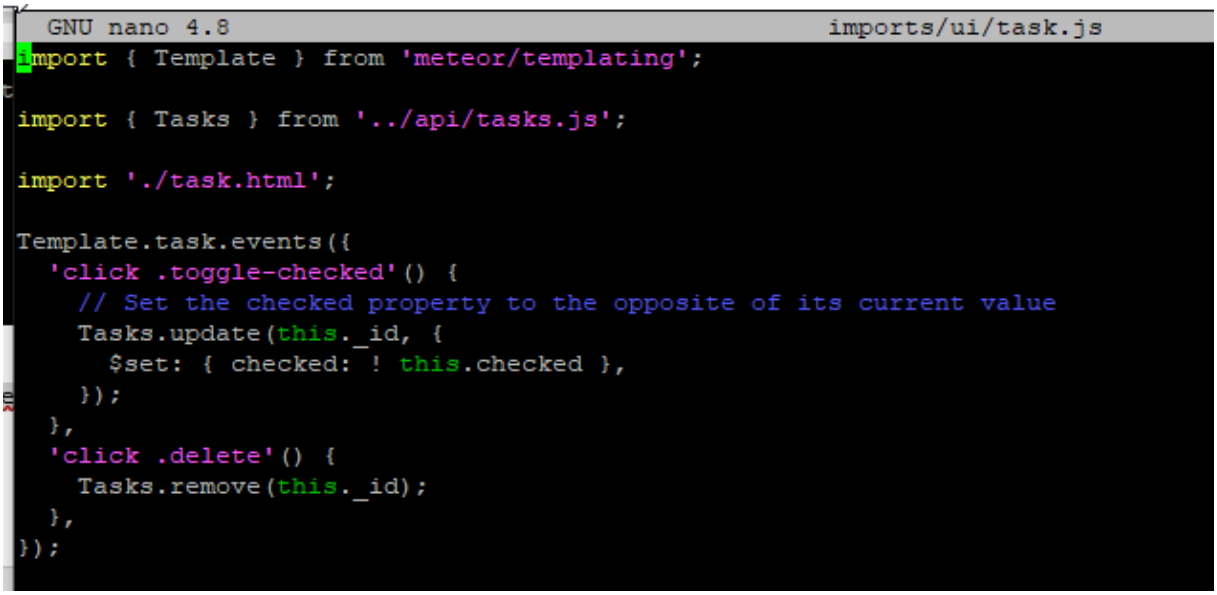
Danach das vorherige „task-template" entfernen:

sudo nano imports/ui/body.html



Danach Events für die task.html erstellen:

sudo nano imports/ui/task.js

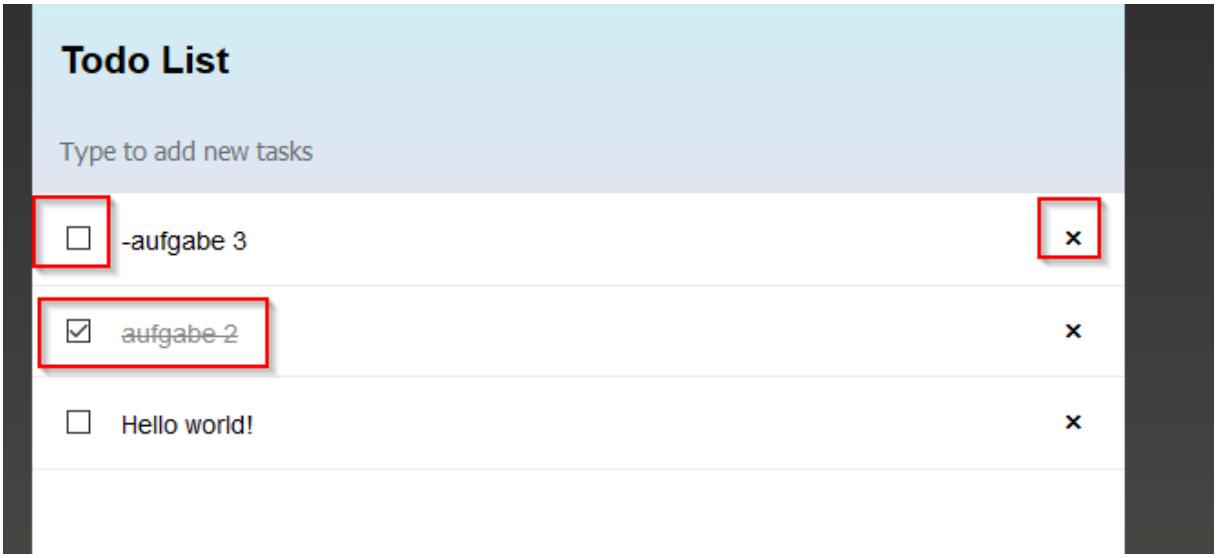


```
import { Template } from 'meteor/templating';

import { Tasks } from '../api/tasks.js';

import './task.html';

Template.task.events({
  'click .toggle-checked'() {
    // Set the checked property to the opposite of its current value
    Tasks.update(this._id, {
      $set: { checked: ! this.checked },
    });
  },
  'click .delete'() {
    Tasks.remove(this._id);
  },
});
```

Danach noch im body.js auf das eben erstellte task.js verweisen:

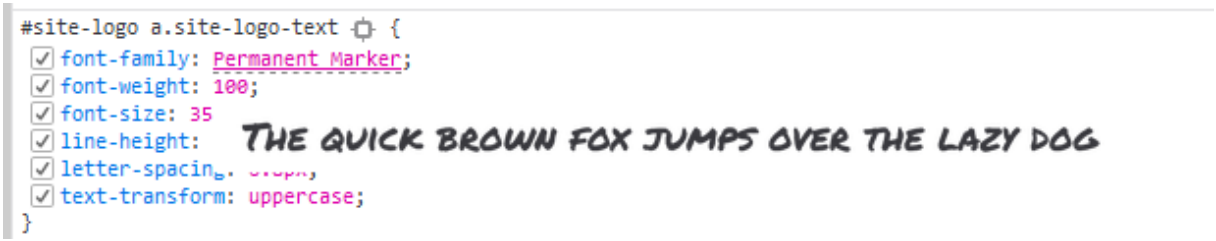
```
import './task.js';
```

Kontrolle:

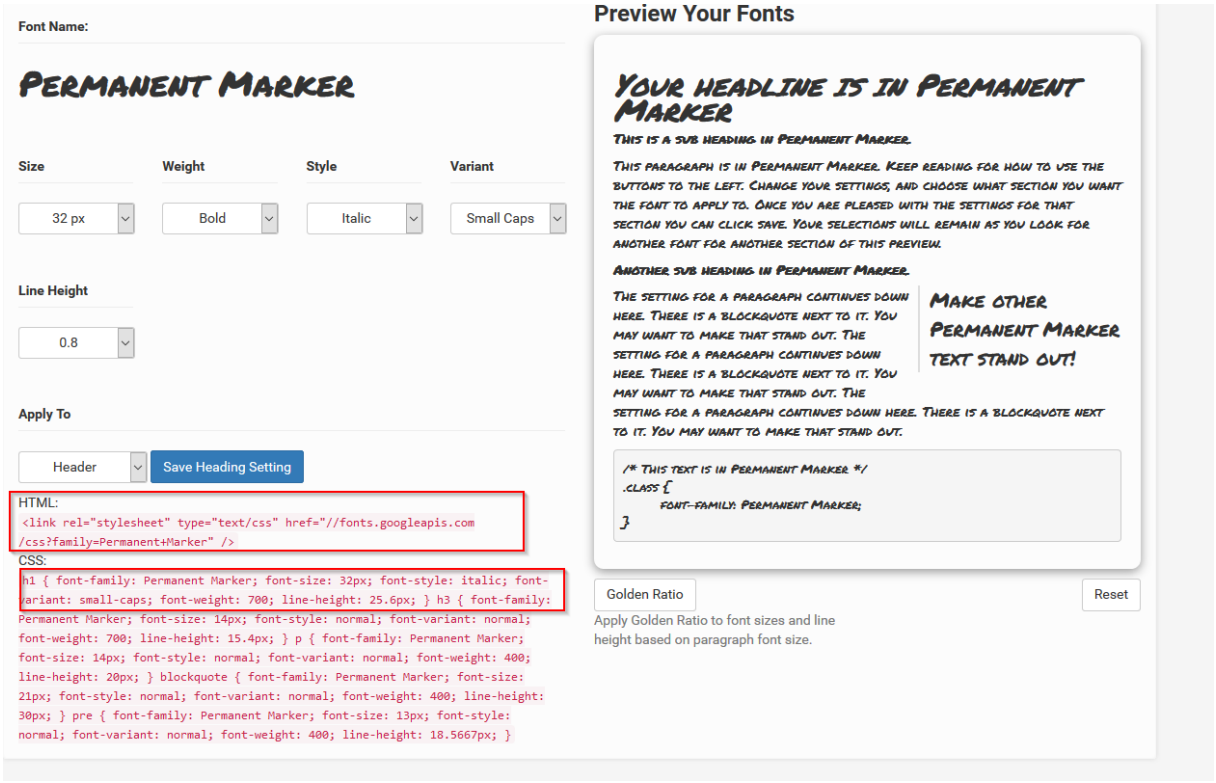


Logo anpassen:

Aus der Homepage von Aufgabe 1 die css Eigenschaften auslesen



Danach auf https://www.cssfontstack.com/Permanent-Marker das gewünschte Syling eingeben bzw. speichern und den link kopieren und im main.html einfügen:



sudo nano client/main.html


```
<link rel="stylesheet" type="text/css"
href="//fonts.googleapis.com/css?family=Permanent+Marker" />
```

Danach body.html bearbeiten

sudo nano imports/ui/body.html

```
  GNU nano 4.8                                          imports/ui/body.html
<body>
    <div class="container">
      <header>
        <h1>Vienna Moonshine Destillery</h1>
        <h2>Einkaufsliste:</h2>
        <form class="new-task">
            <input type="text" name="text" placeholder="Type to add new tasks" />
          </form>

      </header>
      <ul>
        {{#each tasks}}
          {{> task}}
        {{/each}}
      </ul>
    </div>
  </body>
```

<h1>Vienna Moonshine Destillery</h1>

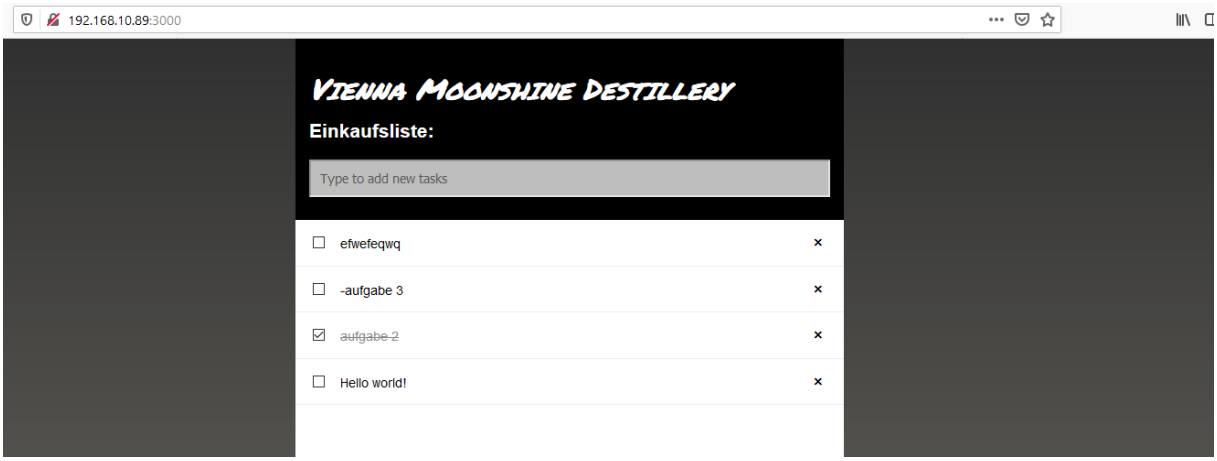
<h2>Einkaufsliste</h2>

Danach noch die css Eigenschaften anpassen, bis sie dem Design der Aufgabe 1 ähneln:

sudo nano client/main.css

```
header {
  background: #d2edf4;
  color: white;
  background-image: linear-gradient(to bottom, #000000, #000000 100%);
  padding: 20px 15px 15px 15px;
  position: relative;
}

#login-buttons {
  display: block;
}

h1 {
  font-family: Permanent Marker;
  font-size: 32px;
  font-style: italic;
  font-variant: small-caps;
  font-weight: 700;
  line-height: 25.6px;
}


h2 {
  font-size: 1.5em;
  margin: 0;
  margin-bottom: 10px;
  display: inline-block;
  margin-right: 1em;
}

form {
  margin-top: 10px;
  margin-bottom: 10px;
  position: relative;
}

.new-task input {
  box-sizing: border-box;
  padding: 10px 10px;
  background: #BEBEBE;
  border: silid white;
  width: 100%;
  padding-right: 80px;
  font-size: 1em;
}
```

Design neu:



Aufgabe 1:

4 PaaS Deployment

Desktop nachinstallieren (Für Heroku CLI wird ein Browser benötigt)

sudo apt-get update -y

sudo apt-get install ubuntu-desktop -y

sudo reboot now

Bei Heroku registrieren/anmelden und App erstellen:

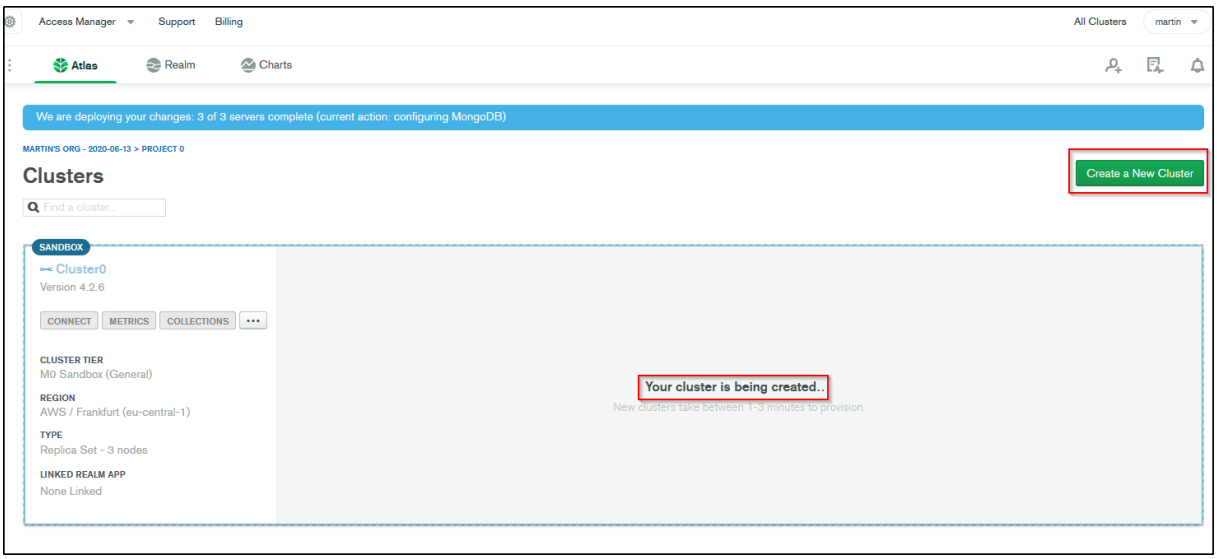
Heroku CLI installieren:

curl https://cli-assets.heroku.com/install-ubuntu.sh | sh

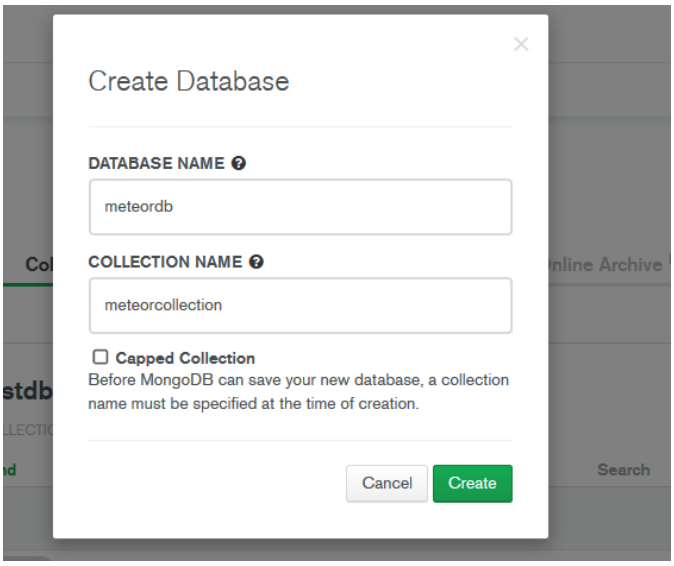
```
martino@meteor:~$ curl https://cli-assets.heroku.com/install-ubuntu.sh | sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1232  100  1232    0     0   4384      0 --:--:-- --:--:-- --:--:--  4384
This script requires superuser access to install apt packages.
You will be prompted for your password by sudo.
[sudo] password for martino:
+ dpkg -s apt-transport-https
+ echo deb https://cli-assets.heroku.com/apt ./
+ dpkg -s heroku-toolbelt
+ true
+ apt-key add -
+ curl https://cli-assets.heroku.com/apt/release.key
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
```

Bei https://www.mongodb.com/cloud/atlas registrieren.

Einen Cluster erstellen:



Eine Datenbank anlegen:



Projektverzeichnis anlegen (optional, dient lediglich der Übersicht)

`mkdir apps`

`cd apps`

Github Repository herunterladen

`git clone https://github.com/meteor/todos.git`

```
martino@meteor:~$ mkdir apps
martino@meteor:~$ cd apps
martino@meteor:~/apps$ git clone https://github.com/meteor/todos.git
Cloning into 'todos'...
remote: Enumerating objects: 36, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 4814 (delta 9), reused 19 (delta 4), pack-reused 4778
Receiving objects: 100% (4814/4814), 1.19 MiB | 1.86 MiB/s, done.
Resolving deltas: 100% (2904/2904), done.
martino@meteor:~/apps$ cd todos/
martino@meteor:~/apps/todos$ meteor npm install

> bcrypt@5.0.0 install /home/martino/apps/todos/node_modules/bcrypt
> node-pre-gyp install --fallback-to-build

node-pre-gyp WARN Using request for node-pre-gyp https download
[bcrypt] Success: "/home/martino/apps/todos/node_modules/bcrypt/lib/binding/napi-v3/bcrypt_lib.node" is installed via remote

> core-js@2.6.11 postinstall /home/martino/apps/todos/node_modules/core-js
> node -e "try{require('./postinstall')}catch(e){}"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilling JavaScript standard library!

The project needs your help! Please consider supporting of core-js on Open Collective or Patreon:
> https://opencollective.com/core-js
> https://www.patreon.com/zloirock

Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job -)


> electron@2.0.18 postinstall /home/martino/apps/todos/node_modules/electron
> node install.js

npm WARN todos No repository field.
npm WARN todos No license field.

added 468 packages from 590 contributors and audited 458 packages in 16.279s

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

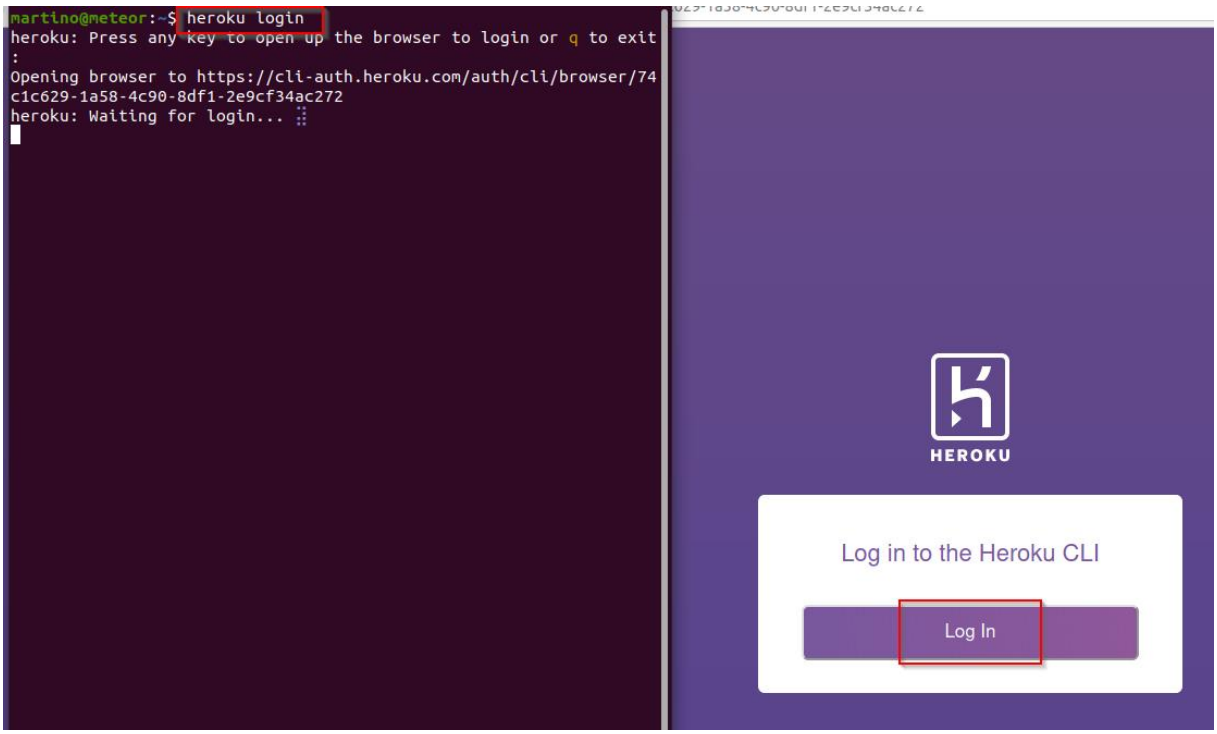
git init

git add .

git commit -m „text“

Mit der Heroku CLI forfahren:

heroku login (danach im Browser den Login verifizieren)



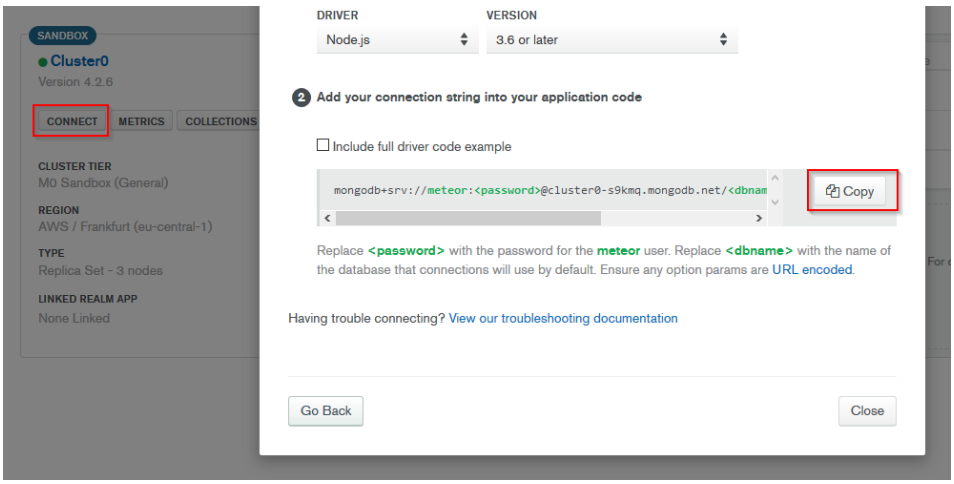
Eine App erstellen:

heroku apps:create Appname

Buildpack installieren, da Heroku meteor nicht unterstützt:

heroku buildpacks:set https://github.com/AdmitHub/meteor-buildpack-horse.git

Danach den Connectionstring für die Datenbank festlegen:



heroku config:add MONGO_URL= mongodb+srv://<DBUser>:<password>@cluster0-s9kmq.mongodb.net/<dbname>?retryWrites=true&w=majority

Für <DBUser> den Datenbankuser verwenden, <passwort> das jeweilige Passwort des Datenbankusers und für <dbname> den Namen der erstellten Datenbank verwenden.

Die Url festlegen:

heroku config:add ROOT_URL=https://<AppName>.herokuapp.com

Deploy der App:

git push heroku master

```
martino@meteor:~/apps/todos$ git init
Reinitialized existing Git repository in /home/martino/apps/todos/.git/
martino@meteor:~/apps/todos$ git add .
martino@meteor:~/apps/todos$ git commit -m "versuch 3"
[master 33fa2c6] versuch 3
 1 file changed, 62 insertions(+)
martino@meteor:~/apps/todos$ heroku login
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/b1306dbd-13cc-49d4-b6cf-10c0be992c4b
Logging in... done
Logged in as martin.andreas@gmx.at
martino@meteor:~/apps/todos$ heroku apps:create foobar222
Creating ⬢ foobar222... done
https://foobar222.herokuapp.com/ | https://git.heroku.com/foobar222.git
martino@meteor:~/apps/todos$ heroku buildpacks:set https://github.com/AdmitHub/meteor-buildpack-horse.git
Buildpack set. Next release on foobar222 will use https://github.com/AdmitHub/meteor-buildpack-horse.git.
Run git push heroku master to create a new release using this buildpack.
martino@meteor:~/apps/todos$ heroku config:add MONGO_URL=mongodb+srv://meteor:[redacted]@cluster0-s9kmq.mongodb.net/meteordb?r
etryWrites=true&w=majority
[1] 5649
Setting MONGO_URL and restarting ⬢ foobar222... done, v3
MONGO_URL: mongodb+srv://meteor:[redacted]@cluster0-s9kmq.mongodb.net/meteordb?retryWrites=true

[1]+  Done                    heroku config:add MONGO_URL=mongodb+srv://meteor:LisaBerger91@cluster0-s9kmq.mongodb.net/meteordb?
retryWrites=true
martino@meteor:~/apps/todos$ heroku config:add ROOT_URL=https://foobar222.herokuapp.com
Setting ROOT_URL and restarting ⬢ foobar222... done, v4
ROOT_URL: https://foobar222.herokuapp.com
martino@meteor:~/apps/todos$ git remote -v
heroku  https://git.heroku.com/foobar222.git (fetch)
heroku  https://git.heroku.com/foobar222.git (push)
origin  https://github.com/meteor/todos.git (fetch)
origin  https://github.com/meteor/todos.git (push)
martino@meteor:~/apps/todos$ git push heroku master
Enumerating objects: 3304, done.
Counting objects: 100% (3304/3304), done.
Compressing objects: 100% (1173/1173), done.
Writing objects: 100% (3304/3304), 820.71 KiB | 31.57 MiB/s, done.
Total 3304 (delta 1988), reused 3298 (delta 1984)
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Node.js app detected
remote: -----> Installing meteor
remote: Downloading Meteor distribution
```

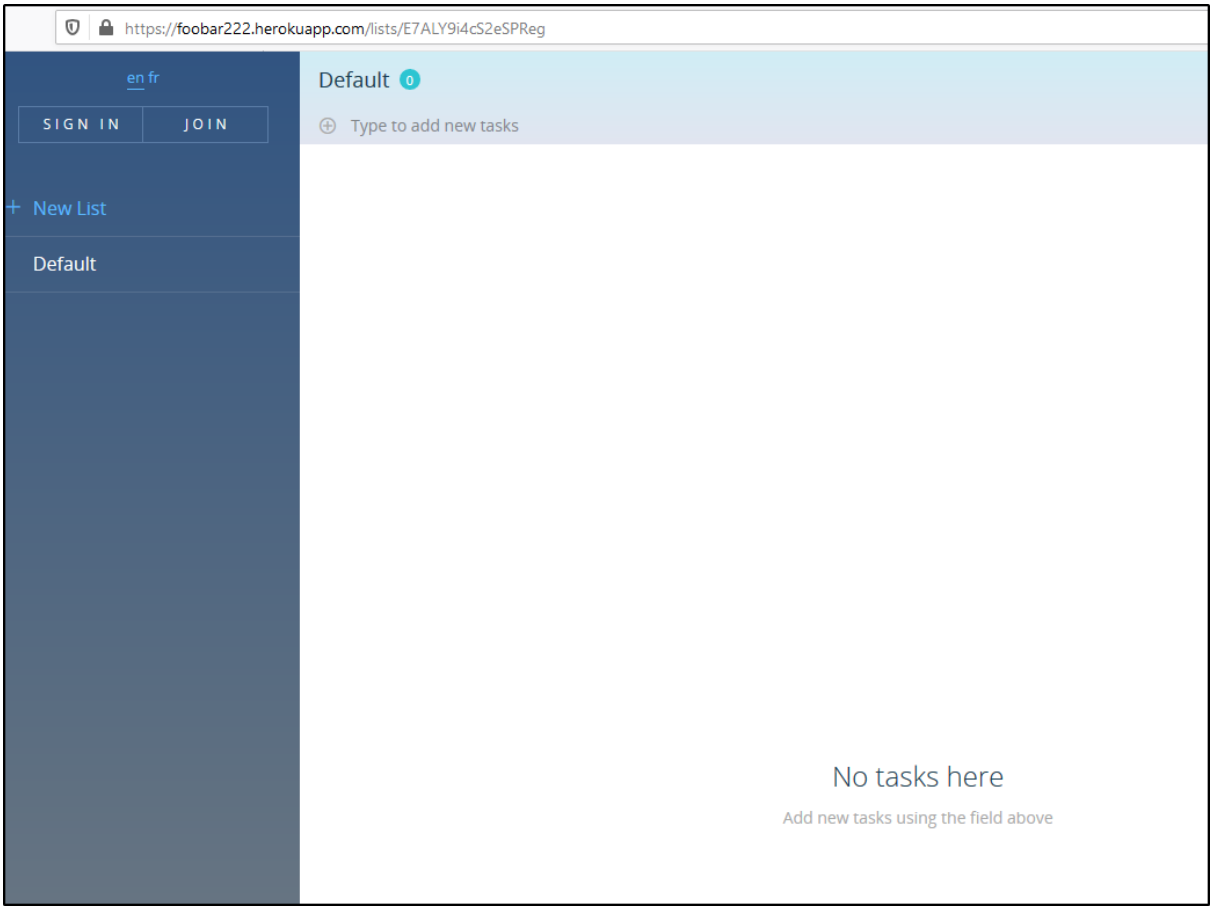
Deployment kann ein paar Minuten dauern!

```
remote: found 0 vulnerabilities
remote:
remote: -----> Adding PATH environment
remote: -----> Running extras
remote: -----> Adding profile script to resolve MONGO_URL from mongolab addon
remote: -----> Adding profile script to resolve ROOT_URL from heroku app name
remote: -----> Discovering process types
remote:        Procfile declares types     -> (none)
remote:        Default types for buildpack -> web
remote:
remote: -----> Compressing...
remote:        Done: 141M
remote: -----> Launching...
remote:        Released v5
remote:        https://foobar222.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/foobar222.git
 * [new branch]      master -> master
martino@meteor:~/apps/todos$
```

App:

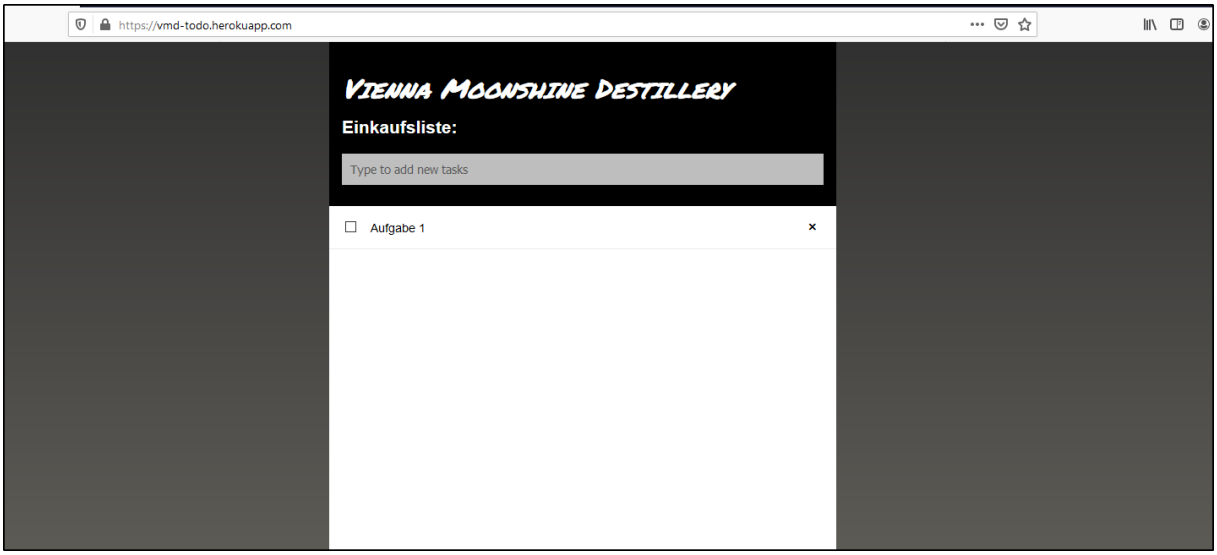
Standard:

https://foobar222.herokuapp.com



Eigenes Design:

https://vmd-todo.herokuapp.com/



5 Ergänzungen

5.1 Sprache hinzufügen

cd i18n/

touch de.i18n.json

cat en.i18n.json > de.i18n.json

sudo nano de.i18.jason

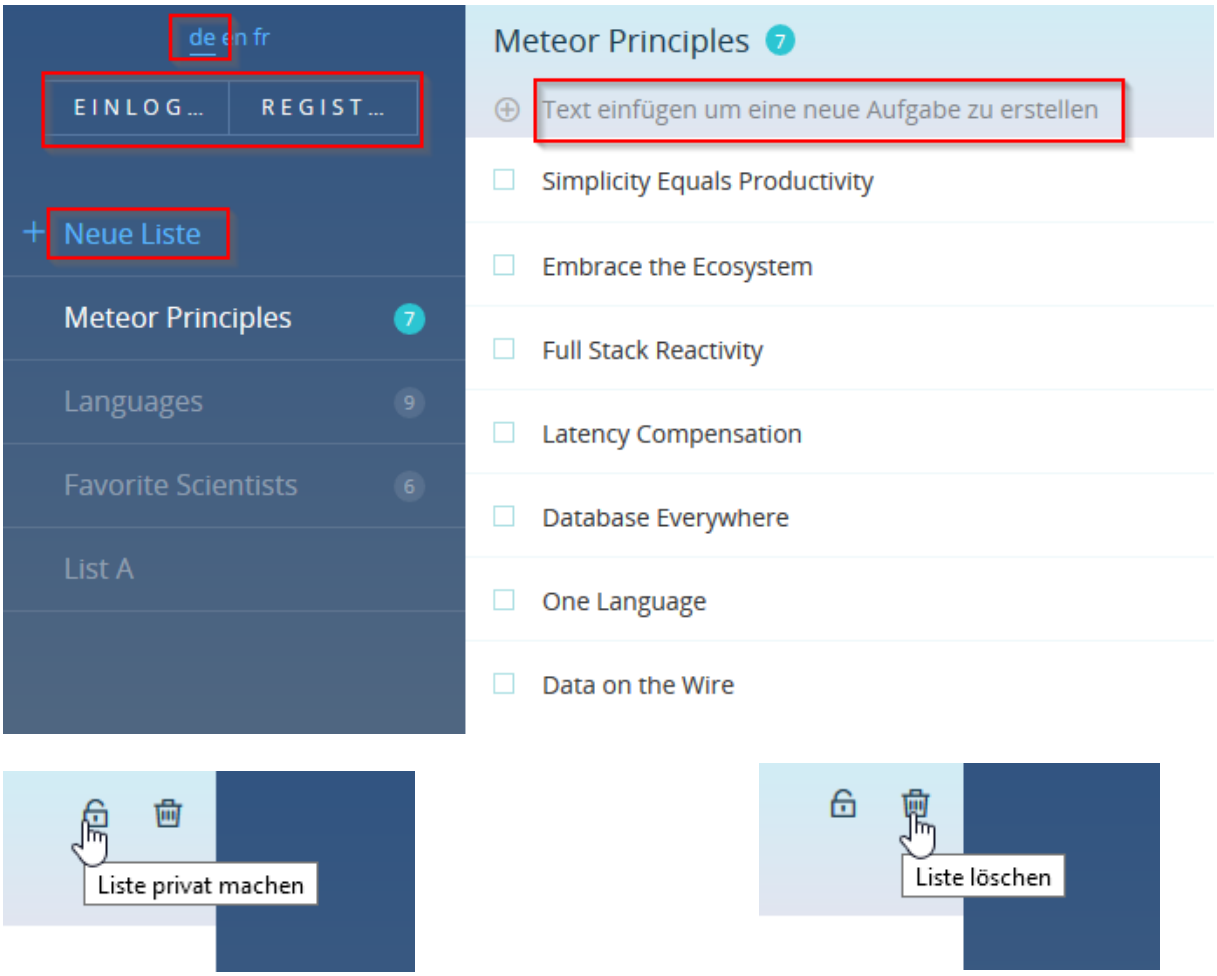
Englische Ausdrücke übersetzen

Beispiel:

```
{
  "lists": {
    "makePrivate": {
      "notLoggedIn": "Du musst eingeloggt sein um Listen zu erstellen.",
      "lastPublicList": "Du kannst die letzte öffentliche Liste nicht auf
privat änderne."
    },
    "makePublic": {
      "notLoggedIn": "Du musst eingeloggt sein.",
      "accessDenied": "Du hast keine Berechtigung um diese Liste zu
bearbeitent."
    },
    "updateName": {
      "accessDenied": "Du hast keine Berechtigung um diese Liste zu
bearbeitent."
    },
    "remove": {
      "accessDenied": "'Du hast keine Berechtigung um diese Liste zu
löschen.'",
      "lastPublicList": "du kannst die letzte öffentliche Liste nicht
löschen.",
      "confirm": "Bist du sicher diese Liste löschen zu wollen?"
    },
    "show": {
      "cancel": "Abbrechen",
      "showMenu": "Menü anzeigen",
      "selectAction": "Aktion wählen",
      "makePublic": "Öffentlich machen",
      "makePrivate": "Privat machen",
      "delete": "Löschen",
      "makeListPublic": "Liste öffentlich machen",
      "makeListPrivate": "Liste privat machen",
      "deleteList": "Liste löschen",
      "typeToAdd": "Text einfügen um eine neue Aufgabe zu erstellen",
      "noTasks": "Keine Aufgaben",
      "addAbove": "Um eine neue Aufgabe zu erstellen benutze das obere
Feld",
      "loading": "Loading tasks..."
    },
```

```
    "insert": {
      "list": "Liste"
    }
  },
  "todos": {
    "insert": {
      "accessDenied": "Du kannst keine Aufgaben zu einer privaten Liste
hinzufügen, die nicht dir gehört"
    },
    "setCheckedStatus": {
      "accessDenied": "Du kannst keine Aufgaben zu einer privaten Liste
hinzufügen, die nicht dir gehört"
    },
    "updateText": {
      "accessDenied": "Du kannst keine Aufgaben zu einer privaten Liste
hinzufügen, die nicht dir gehört"
    },
    "remove": {
      "accessDenied": "Du kannst keine Aufgaben zu einer privaten Liste
hinzufügen, die nicht dir gehört"
    },
    "item": {
      "taskName": "Bezeichnung der Aufgabe"
    }
  },
  "useraccounts": {
    "atTitle": {
      "subtitle": "Nur eingeloggte User können private Listen erstellen"
    }
  },
  "layouts": {
    "appBody": {
      "logout": "Ausloggen",
      "login": "Einloggen",
      "join": "Registrieren",
      "newList": "Neue Liste",
      "newListError": "Liste konnte nicht erstellt werden.",
      "tryingToConnect": "Verbindung wird hergestelltt",
      "connectionIssue": "Verbindungsprobleme"
    }
  },
  "pages": {
    "appNotFound": {
      "pageNotFound": "Seite konnte nicht gefunden werden"
    }
  }
}
```

Kontrolle der funktionalität:


de en fr
EINLOG...
REGIST...
Neue Liste
Meteor Principles 7
Languages 9
Favorite Scientists 6
List A
Meteor Principles 7
Text einfügen um eine neue Aufgabe zu erstellen
Simplicity Equals Productivity
Embrace the Ecosystem
Full Stack Reactivity
Latency Compensation
Database Everywhere
One Language
Data on the Wire

Liste privat machen

Liste löschen

5.2 Raspberry Pi

5.2.1 Meteor

Plattform: Raspberry Pi 3 Model B

Da Meteor (https://www.meteor.com/install) nicht für eine ARM-Architektur zur Verfügung steht muss für einen Raspberry Pi ein Meteor-fork verwendet werden.

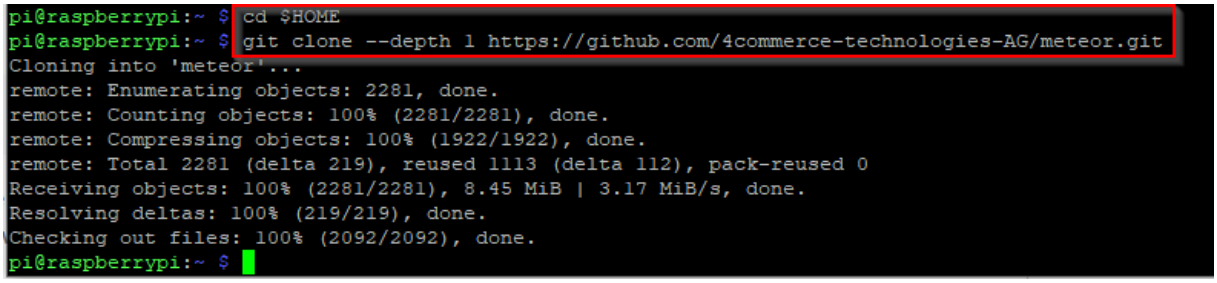
Meteor installieren:

Ins Verzeichnis wechseln, wo Meteor installiert werden soll (Bsp. „Home-Verzeichnis")

```
cd $HOME
```

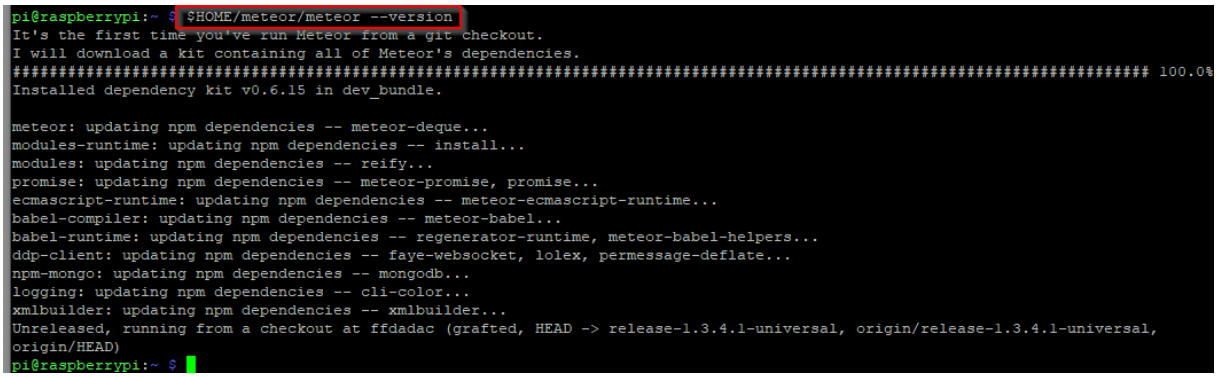
Github Repository herunterladen:

```
git clone --depth 1 https://github.com/4commerce-technologies-AG/meteor.git
```



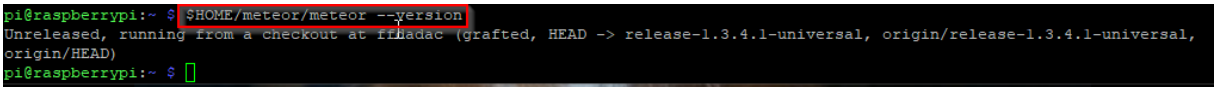
Um die Installation abzuschließen muss ein kompatibles vorgefertigtes „dev_bundle" heruntergeladen werden. Mit demselben Befehl kann man nachträglich auch die Version überprüfen:

```
$HOME/meteor/meteor –version
```



Das kann je nach Hardware zeitlich variieren, im Fall des Raspberry Pi hat es ca. 30 Minuten gedauert.

Version überprüfen:



5.2.2 Todo-Applikation

Todo App installieren:

```
$HOME/meteor/meteor create --example todos
```

```
git clone https://github.com/meteor/todos
```

```
pi@raspberrypi:~/meteor $ $HOME/meteor/meteor create --example todos
To create the todos example, please run:
  git clone https://github.com/meteor/todos
pi@raspberrypi:~/meteor $ git clone https://github.com/meteor/todos
Cloning into 'todos'...
remote: Enumerating objects: 350, done.
remote: Counting objects: 100% (350/350), done.
remote: Compressing objects: 100% (178/178), done.
remote: Total 4776 (delta 235), reused 264 (delta 169), pack-reused 4426
Receiving objects: 100% (4776/4776), 1.16 MiB | 1.89 MiB/s, done.
Resolving deltas: 100% (2878/2878), done.
```

Danach in das Projektverzeichnis wechseln:

```
cd todos
```

Und npm installieren

```
$HOME/meteor/meteor npm install
```

- npm installieren konnte nicht durchgeführt werden. Es wurden mehrere Errors und Warnings identifiziert (ein Aausschnitt davon im Bild)

```
npm WARN ajv-keywords@2.1.1 requires a peer of ajv@^5.0.0 but none was installed.
npm WARN eslint-config-airbnb@11.2.0 requires a peer of eslint@^3.6.0 but none was installed.
npm WARN eslint-config-airbnb-base@7.2.0 requires a peer of eslint@^3.6.0 but none was installed.
npm WARN eslint-plugin-import@1.16.0 requires a peer of eslint@2.x - 3.x but none was installed.
npm WARN eslint-plugin-jsx-ally@2.2.3 requires a peer of eslint@^2.10.2 || 3.x but none was installed.
npm WARN eslint-plugin-react@6.10.3 requires a peer of eslint@^2.0.0 || ^3.0.0 but none was installed.
npm WARN todos No repository field.
npm WARN todos No license field.
npm ERR! Linux 4.19.97-v7+
npm ERR! argv "node" "/home/pi/meteor/dev_bundle/bin/npm" "install"
npm ERR! node v0.10.45
npm ERR! npm  v3.9.6
npm ERR! code ELIFECYCLE

npm ERR! electron@2.0.18 postinstall: `node install.js`
npm ERR! Exit status 8
npm ERR!
npm ERR! Failed at the electron@2.0.18 postinstall script 'node install.js'.
npm ERR! Make sure you have the latest version of node.js and npm installed.
npm ERR! If you do, this is most likely a problem with the electron package,
npm ERR! not with npm itself.
npm ERR! Tell the author that this fails on your system:
npm ERR!     node install.js
npm ERR! You can get information on how to open an issue for this project with:
npm ERR!     npm bugs electron
npm ERR! Or if that isn't available, you can get their info via:
npm ERR!     npm owner ls electron
npm ERR! There is likely additional logging output above.

npm ERR! Please include the following file with any support request:
npm ERR!     /home/pi/meteor/todos/npm-debug.log
```

Die Todo-App gibt es auch als reactJS (eigentlich eine Ergänzungsaufgabe, hier jedoch notwendig, um die Todo-App zu erstellen):

```
cd $HOME
```

$HOME/meteor/meteor create --example simple-todos-react

git clone https://github.com/meteor/simple-todos-react

cd simple-todos-react

$HOME/meteor/meteor npm install

```
pi@raspberrypi:~/meteor/todos $ cd $HOME
pi@raspberrypi:~ $ $HOME/meteor/meteor create --example simple-todos-react
To create the simple-todos-react example, please run:
  git clone https://github.com/meteor/simple-todos-react
pi@raspberrypi:~ $ git clone https://github.com/meteor/simple-todos-react
Cloning into 'simple-todos-react'...
remote: Enumerating objects: 241, done.
remote: Counting objects: 100% (241/241), done.
remote: Compressing objects: 100% (134/134), done.
remote: Total 1437 (delta 105), reused 226 (delta 97), pack-reused 1196
Receiving objects: 100% (1437/1437), 509.91 KiB | 1.53 MiB/s, done.
Resolving deltas: 100% (685/685), done.
pi@raspberrypi:~ $ cd simple-todos-react
pi@raspberrypi:~/simple-todos-react $ $HOME/meteor/meteor npm install
simple-todos-react@ /home/pi/simple-todos-react
├─┬ @babel/runtime@7.9.6
│ └── regenerator-runtime@0.13.5
├─┬ chai@4.2.0
│ └── assertion-error@1.1.0
```

Die packages aktualisieren:

$HOME/meteor/meteor update --packages-only

- ergab ein paar Fehlermeldungen!

```
pi@raspberrypi:~/simple-todos-react $ $HOME/meteor/meteor update --packages-only
=> Errors while upgrading packages:

While selecting package versions:
error: No version of meteor-base satisfies all constraints: @1.4.0, @=1.0.4
Constraints on package "meteor-base":
* meteor-base@1.4.0 <- top level
* meteor-base@=1.0.4 <- top level

No version of mobile-experience satisfies all constraints: @1.0.5, @=1.0.4
Constraints on package "mobile-experience":
* mobile-experience@1.0.5 <- top level
```

Ein Package hinzufügen, um zu vermeiden, dass die Modulnutzung und die DDP-Fehlermeldung beim Start gezählt werden:

$HOME/meteor/meteor add package-stats-opt-out

- ergab ebenfalls ein paar Fehler!

```
pi@raspberrypi:~/simple-todos-react $ $HOME/meteor/meteor add package-stats-opt-out
=> Running Meteor from a checkout -- overrides project version (Meteor 1.8.0.2)
 => Errors while adding packages:

While selecting package versions:
error: No version of meteor-base satisfies all constraints: @1.4.0, @=1.0.4
Constraints on package "meteor-base":
* meteor-base@1.4.0 <- top level
* meteor-base@=1.0.4 <- top level

No version of mobile-experience satisfies all constraints: @1.0.5, @=1.0.4
Constraints on package "mobile-experience":
* mobile-experience@1.0.5 <- top level
```

```
No version of reactive-var satisfies all constraints: @1.0.11, @=1.0.10
Constraints on package "reactive-var":
* reactive-var@1.0.11 <- top level
* reactive-var@=1.0.10 <- top level

No version of tracker satisfies all constraints: @1.2.0, @=1.0.14
Constraints on package "tracker":
* tracker@1.2.0 <- top level
* tracker@=1.0.14 <- top level
* tracker@1.0.10-beta.16 <- react-meteor-data 0.2.6-beta.16

No version of standard-minifier-css satisfies all constraints: @1.5.2, @=1.0.7_1
Constraints on package "standard-minifier-css":
* standard-minifier-css@1.5.2 <- top level
* standard-minifier-css@=1.0.7_1 <- top level

No version of standard-minifier-js satisfies all constraints: @2.4.0, @=1.0.7_1
Constraints on package "standard-minifier-js":
```

Die App starten:

`$HOME/meteor/meteor`

- Die App konnte nicht gestartet werden, da es Probleme mit MongoDB gab:

```
pi@raspberrypi:~/simple-todos-react $ $HOME/meteor/meteor
=> Running Meteor from a checkout -- overrides project version (Meteor 1.8.0.2)
[[[[[ ~/simple-todos-react ]]]]]

=> Started proxy.
Unexpected mongo exit code 1. Restarting.
Unexpected mongo exit code 1. Restarting.
Unexpected mongo exit code 1. Restarting.
Can't start Mongo server.
MongoDB failed global initialization

Looks like MongoDB doesn't understand your locale settings. See
https://github.com/meteor/meteor/issues/4019 for more details.
```

5.2.3 Problembehandlung:

Abbruch, da das Repository veraltet ist (2014) und die Abhängigkeiten dadurch nicht mehr stimmen.