IoT Experimental Learning
Week 5 Journal

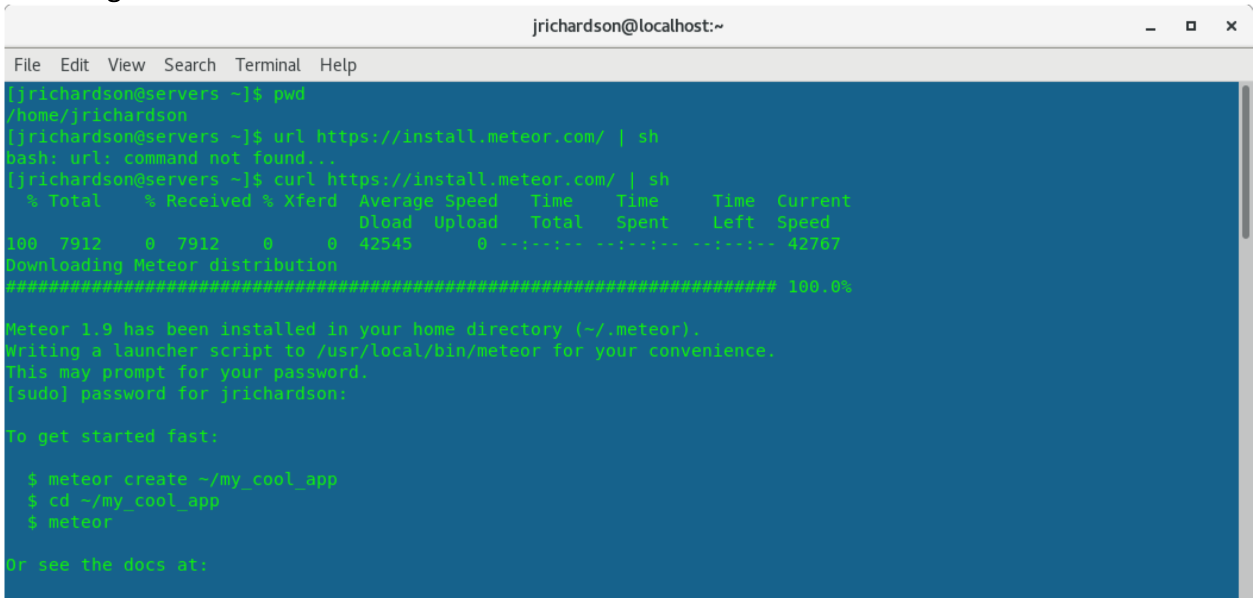
1. **Describe the experience and what you hope to gain from participating in the experience.**
    - This week's assignment was a very easy for me, I found no issues while trying to complete assignment.
        1. Reviewed weekly assignment material.
        2. Completed Coding assignment
        3. Tested written code

2. **Provide an overview of tasks and key activities (training, discussions, labs, assessments, etc.) in which you were engaged during the week.**

    For week 5 I accomplished the following tasks in chronological order;

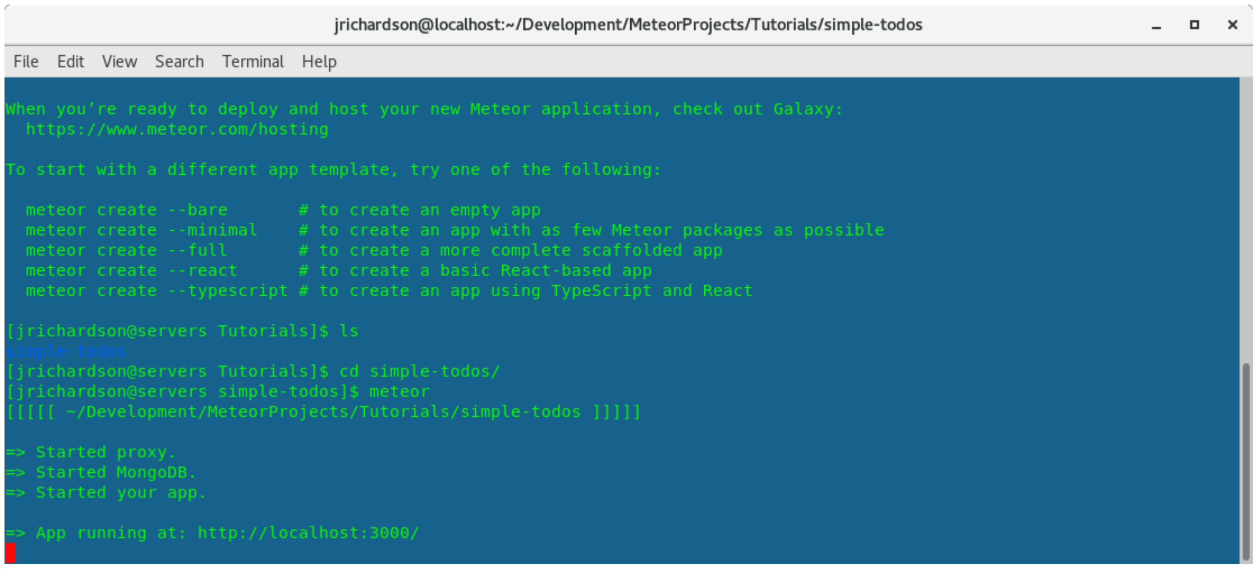
❖ **Monday February 17, 2020**, completed reviewing this week's assignment for the project.
- ➢ I reviewed the Module 4 weekly assignment on the SNHU Brightspace. https://learn.snhu.edu/d2l/le/content/343560/Home
- ➢ Installed Meteor on local CentOS Linux machine
- ➢ Installed Mongo Docker Image to run mongo as container
- ➢ Completed the Meteor Simple ToDos Tutorial at:
    - ▪ https://www.meteor.com/tutorials/blaze/creating-an-app

- ➢ Installing Meteor



```
[jrichardson@servers ~]$ pwd
/home/jrichardson
[jrichardson@servers ~]$ url https://install.meteor.com/ | sh
bash: url: command not found...
[jrichardson@servers ~]$ curl https://install.meteor.com/ | sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  7912    0  7912    0     0  42545      0 --:--:-- --:--:-- --:--:-- 42767
Downloading Meteor distribution
######################################################################## 100.0%

Meteor 1.9 has been installed in your home directory (~/.meteor).
Writing a launcher script to /usr/local/bin/meteor for your convenience.
This may prompt for your password.
[sudo] password for jrichardson:

To get started fast:

  $ meteor create ~/my_cool_app
  $ cd ~/my_cool_app
  $ meteor

Or see the docs at:
```

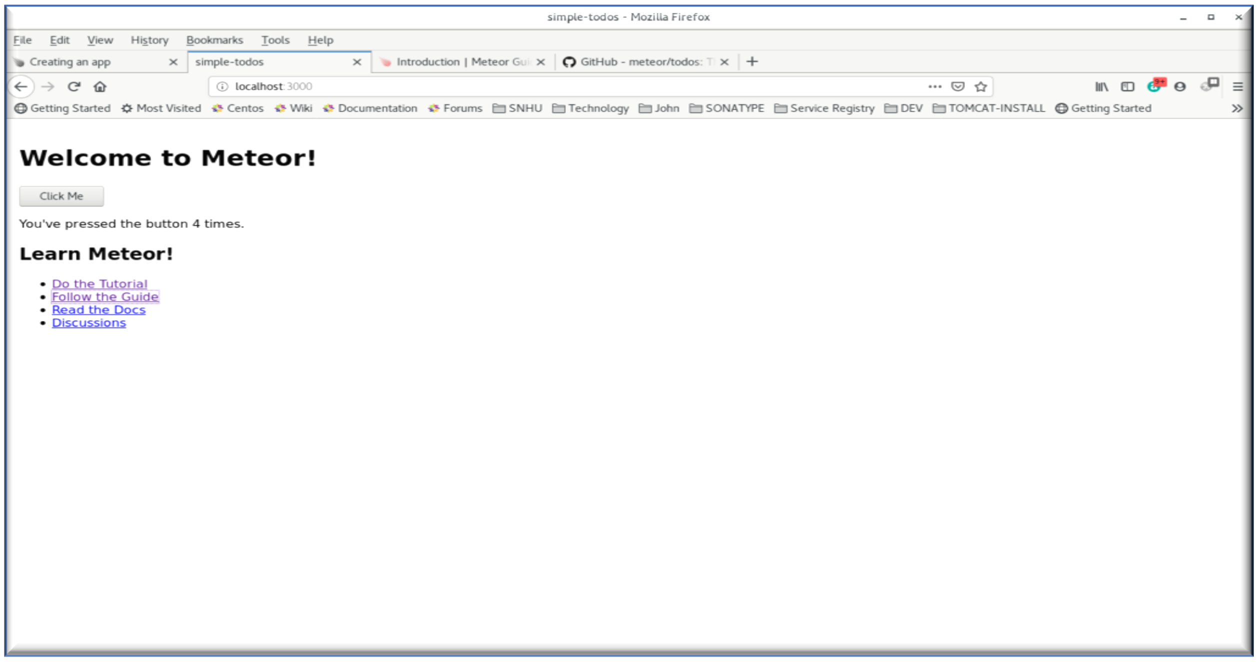
- Run Meteor for the first time



jrichardson@localhost:~/Development/MeteorProjects/Tutorials/simple-todos
File Edit View Search Terminal Help

```
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

[jrichardson@servers Tutorials]$ ls
simple-todos
[jrichardson@servers Tutorials]$ cd simple-todos/
[jrichardson@servers simple-todos]$ meteor
[[[[[ ~/Development/MeteorProjects/Tutorials/simple-todos ]]]]]

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

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

- First Run Client View



simple-todos - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Creating an app
simple-todos
Introduction | Meteor Gu
GitHub - meteor/todos: T
localhost:3000
Getting Started Most Visited Centos Wiki Documentation Forums SNHU Technology John SONATYPE Service Registry DEV TOMCAT-INSTALL Getting Started

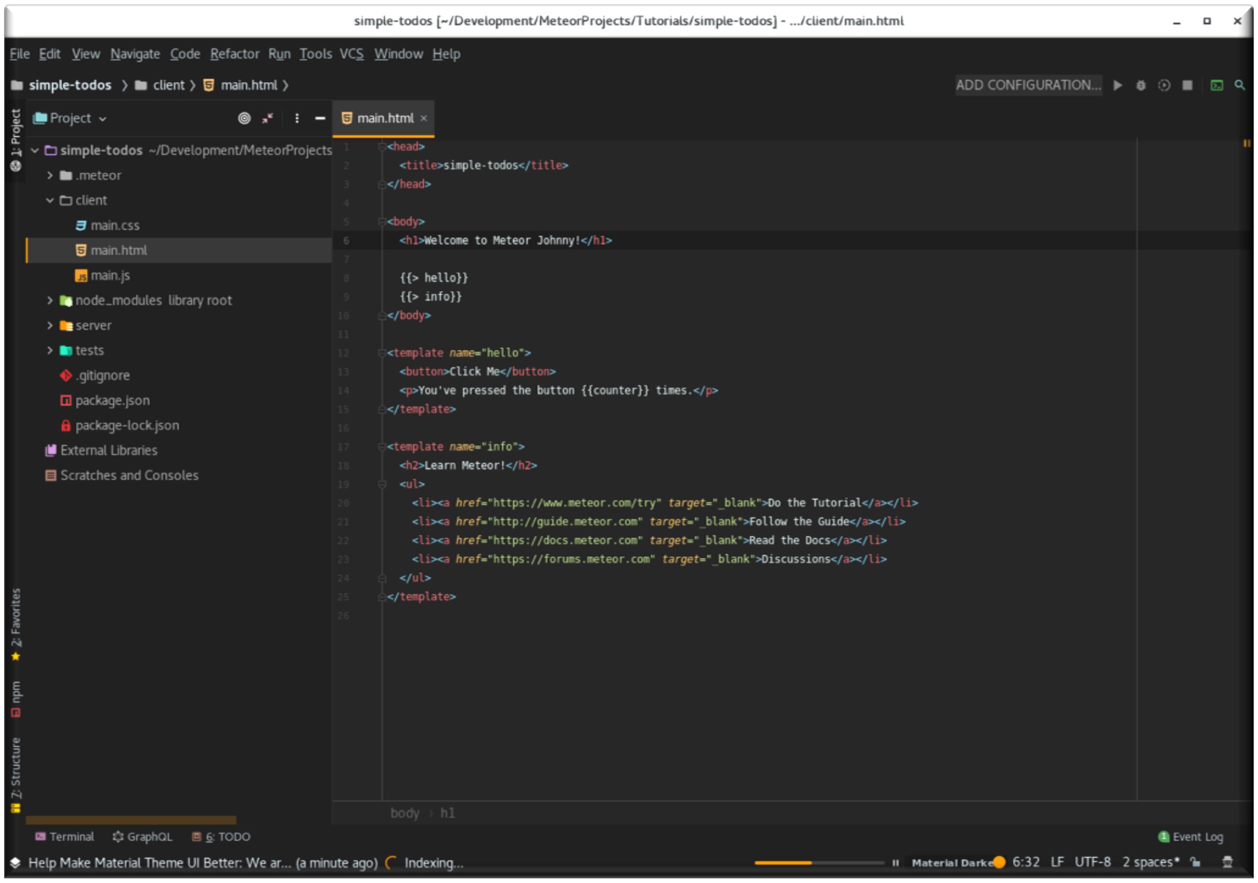
Welcome to Meteor!

You've pressed the button 4 times.

Learn Meteor!

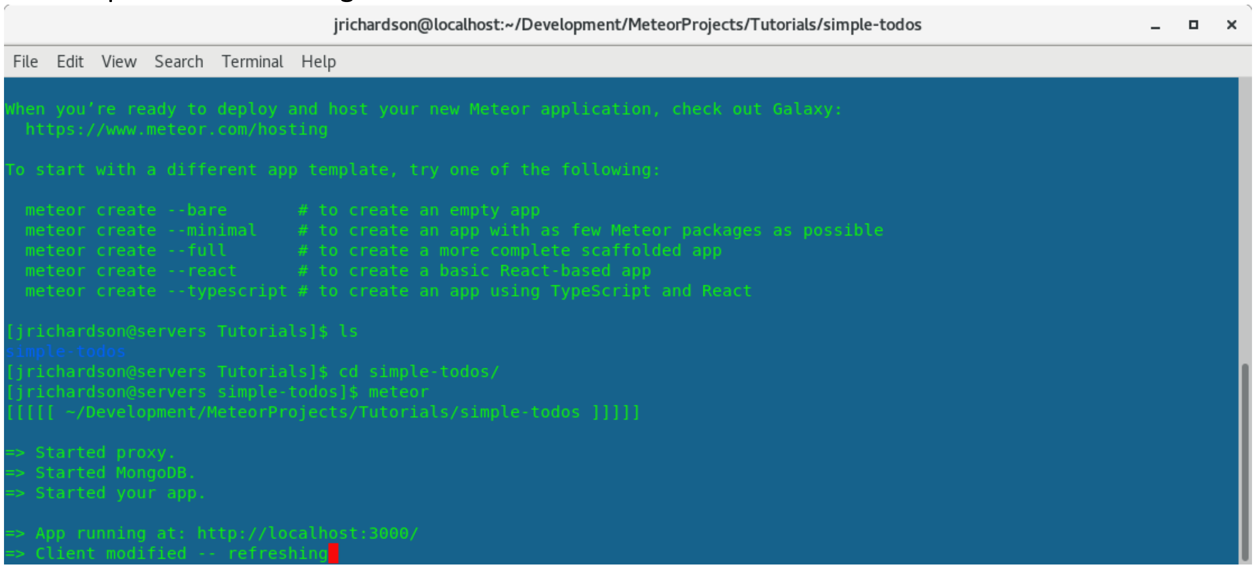
- Do the Tutorial
- Follow the Guide
- Read the Docs
- Discussions

- Added First Mod to Application (Simple)


simple-todos [~/Development/MeteorProjects/Tutorials/simple-todos] - .../client/main.html
File Edit View Navigate Code Refactor Run Tools VCS Window Help
simple-todos › client › main.html ›
ADD CONFIGURATION...
Project
simple-todos ~/Development/MeteorProjects
.meteor
client
main.css
main.html
main.js
node_modules library root
server
tests
.gitignore
package.json
package-lock.json
External Libraries
Scratches and Consoles
body › h1
Terminal
GraphQL
6: TODO
Event Log
Help Make Material Theme UI Better: We ar... (a minute ago)
Indexing...
Material Darker
6:32 LF UTF-8 2 spaces*

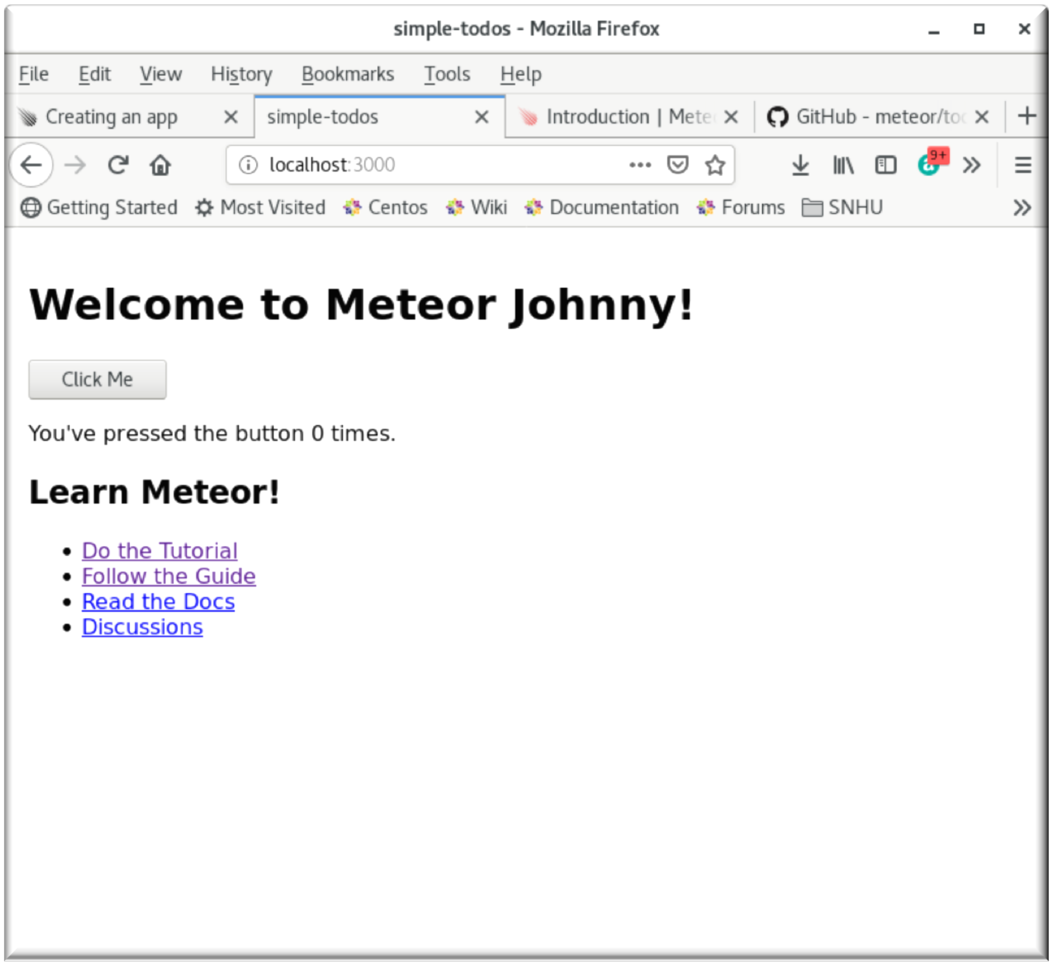
```html
<head>
  <title>simple-todos</title>
</head>

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

- Client Updated from change


jrichardson@localhost:~/Development/MeteorProjects/Tutorials/simple-todos
File Edit View Search Terminal Help

```
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

[jrichardson@servers Tutorials]$ ls
simple-todos
[jrichardson@servers Tutorials]$ cd simple-todos/
[jrichardson@servers simple-todos]$ meteor
[[[[[ ~/Development/MeteorProjects/Tutorials/simple-todos ]]]]]

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

=> App running at: http://localhost:3000/
=> Client modified -- refreshing
```

➢ Change reflected on client ui



Welcome to Meteor Johnny!

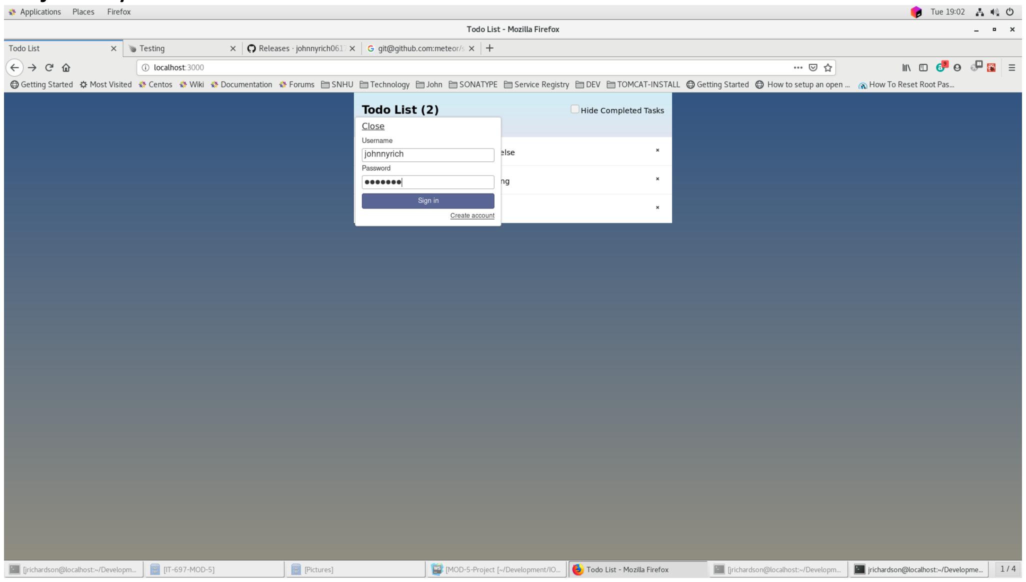
Click Me

You've pressed the button 0 times.

Learn Meteor!

- Do the Tutorial
- Follow the Guide
- Read the Docs
- Discussions

➢ Completed Tutorial Simple Todo
Login: johnnyRich



Todo List (2)

☐ Hide Completed Tasks

Close

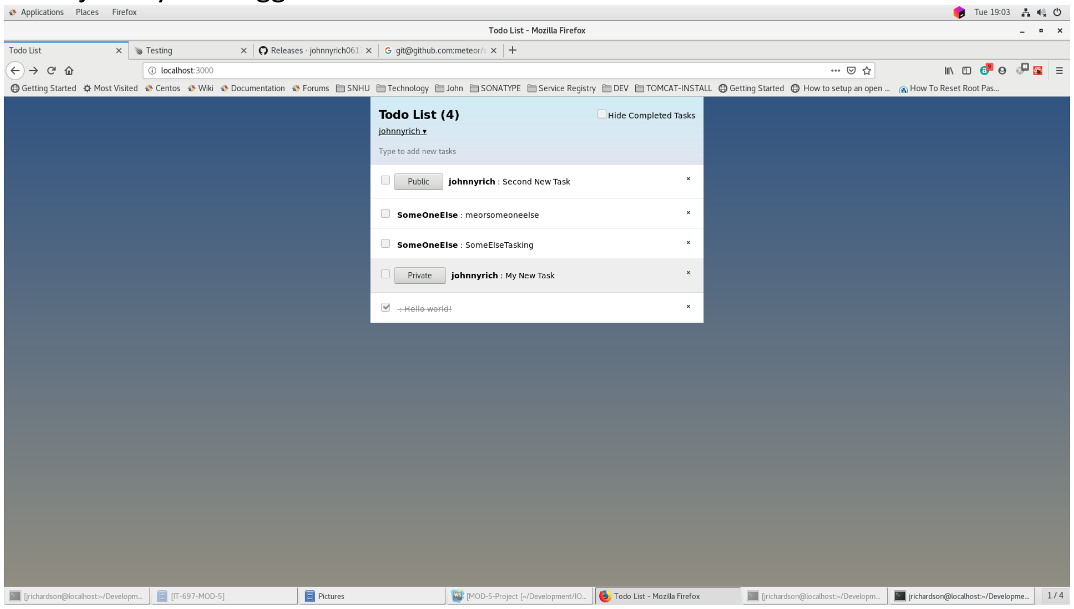
Username: johnnyrich

Password: •••••••

Sign in

Create account

- User: johnnyRich logged in


Applications Places Firefox
Tue 19:03
Todo List - Mozilla Firefox
Todo List
Testing
Releases · johnnyrich061
git@github.com:meteor/
localhost:3000
Getting Started Most Visited Centos Wiki Documentation Forums SNHU Technology John SONATYPE Service Registry DEV TOMCAT-INSTALL Getting Started How to setup an open ... How To Reset Root Pas...

Todo List (4)

Hide Completed Tasks

johnnyrich ▾

Type to add new tasks

Public **johnnyrich** : Second New Task

SomeOneElse : meorsomeoneelse

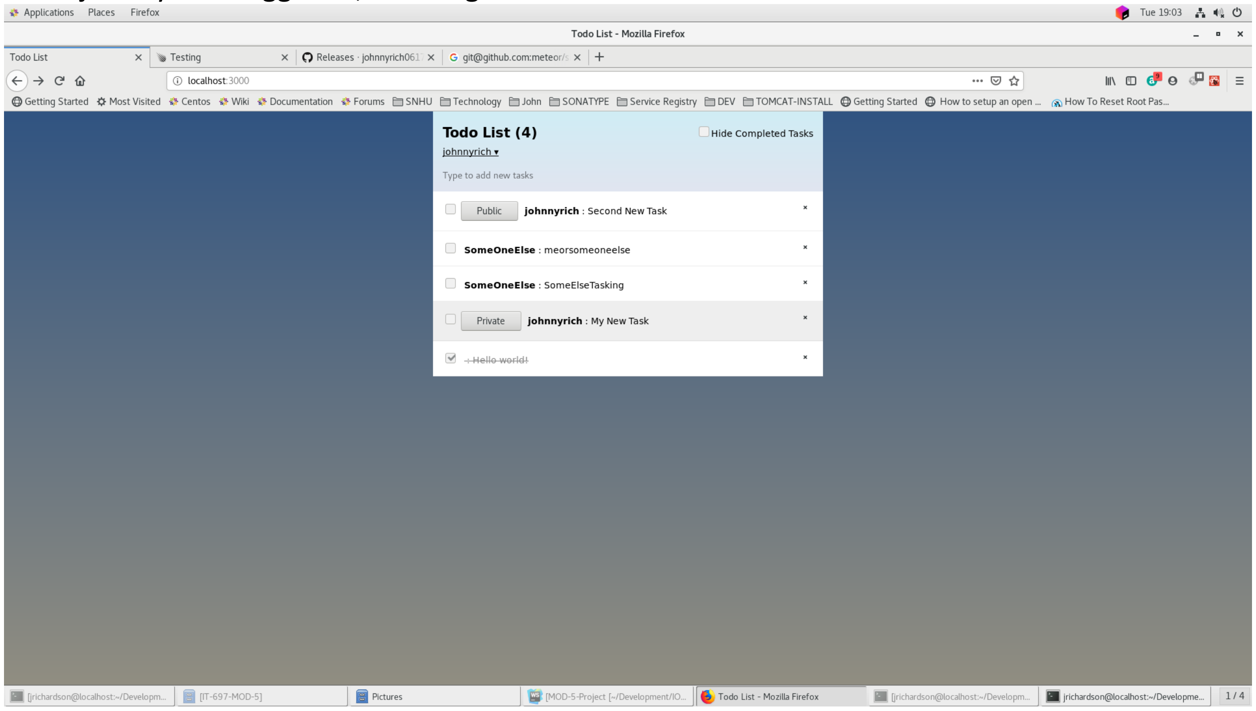
SomeOneElse : SomeElseTasking

Private **johnnyrich** : My New Task

~~: Hello world!~~

[jrichardson@localhost:~/Developm...
[IT-697-MOD-5]
Pictures
[MOD-5-Project [~/Development/IO...
Todo List - Mozilla Firefox
[jrichardson@localhost:~/Developm...
jrichardson@localhost:~/Developme...
1 / 4

User: johnnyRich Logged in, showing Public and Private with selection buttons


Applications Places Firefox
Tue 19:03
Todo List - Mozilla Firefox
Todo List
Testing
Releases · johnnyrich061
git@github.com:meteor/
localhost:3000
Getting Started Most Visited Centos Wiki Documentation Forums SNHU Technology John SONATYPE Service Registry DEV TOMCAT-INSTALL Getting Started How to setup an open ... How To Reset Root Pas...

Todo List (4)

Hide Completed Tasks

johnnyrich ▾

Type to add new tasks

Public **johnnyrich** : Second New Task

SomeOneElse : meorsomeoneelse

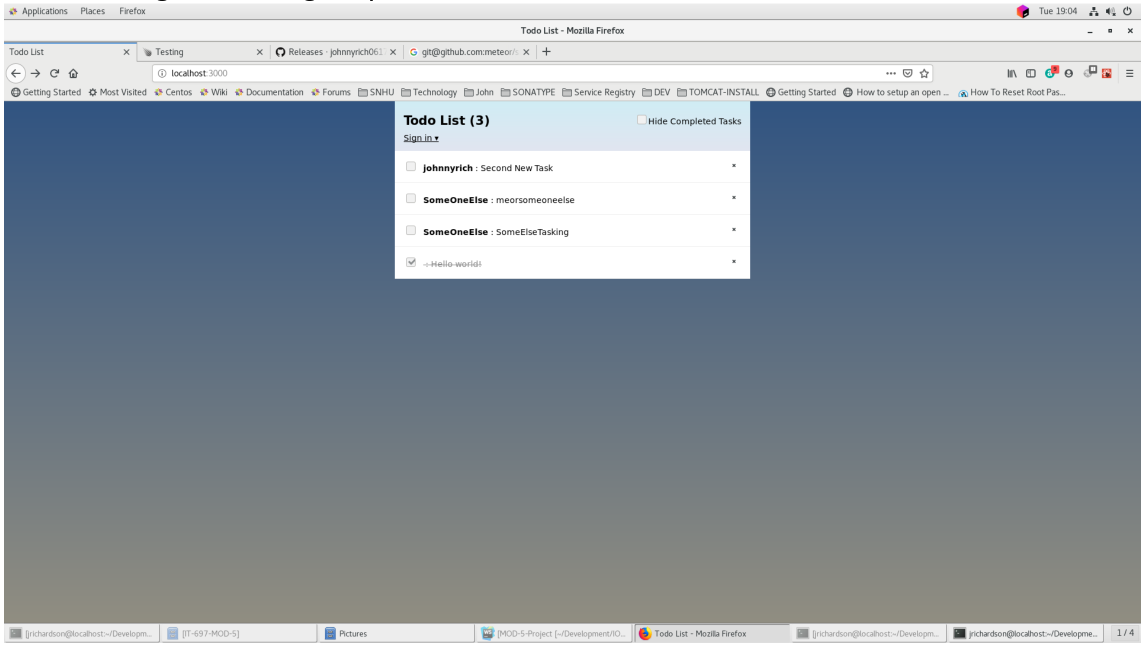
SomeOneElse : SomeElseTasking

Private **johnnyrich** : My New Task

~~: Hello world!~~

[jrichardson@localhost:~/Developm...
[IT-697-MOD-5]
Pictures
[MOD-5-Project [~/Development/IO...
Todo List - Mozilla Firefox
[jrichardson@localhost:~/Developm...
jrichardson@localhost:~/Developme...
1 / 4

User: No Login, showing only Public entities



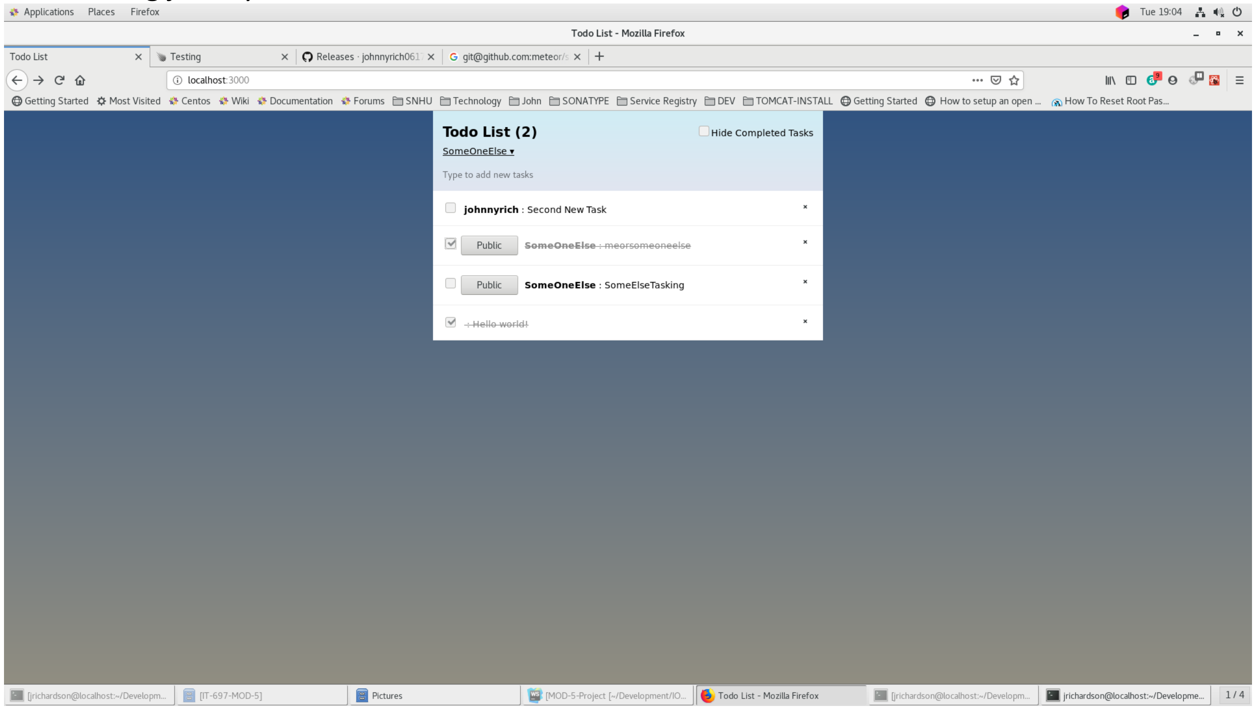
Todo List (3)

Hide Completed Tasks

Sign in ▾

- [ ] **johnnyrich** : Second New Task ×
- [ ] **SomeOneElse** : meorsomeoneelse ×
- [ ] **SomeOneElse** : SomeElseTasking ×
- [✓] ~~: Hello world!~~ ×

➢ User: SomeOneElse Logged in, showing allowed entities and controls for user, only showing johnnyRich Public entities



Todo List (2)

Hide Completed Tasks

SomeOneElse ▾

Type to add new tasks

- [ ] **johnnyrich** : Second New Task ×
- [✓] Public ~~**SomeOneElse** : meorsomeoneelse~~ ×
- [ ] Public **SomeOneElse** : SomeElseTasking ×
- [✓] ~~: Hello world!~~ ×

➢ Client Meteor Code

Main.css

```css
/* 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;
}
```

```css
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;
  }
}
```

Main.html

```html
<head>
    <title>Todo List</title>
</head>

<body>
    <div id="render-target"></div>
</body>
```

Main.js

```
import React from 'react';
import { Meteor } from 'meteor/meteor';
import { render } from 'react-dom';
import '../imports/startup/accounts-config.js';
import App from '../imports/ui/App.js';

Meteor.startup(() => {
  render(<App />, document.getElementById('render-target'));
});
```

tasks.js

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

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

if (Meteor.isServer) {
    // This code only runs on the server
    // Only publish tasks that are public or belong to the current user
    Meteor.publish('tasks', function tasksPublication() {
        return Tasks.find({
            $or: [
                { private: { $ne: true } },
                { owner: this.userId },
            ],
        });
    });
}

Meteor.methods({
    'tasks.insert'(text) {
        check(text, String);
        // Make sure the user is logged in before inserting a task
        if (! this.userId) {
            throw new Meteor.Error('not-authorized');
        }

        Tasks.insert({
            text,
            createdAt: new Date(),
            owner: this.userId,
            username: Meteor.users.findOne(this.userId).username,
        });
    },

    'tasks.remove'(taskId) {
        check(taskId, String);

        const task = Tasks.findOne(taskId);
       // if there is no logged in user then return
        if(!this.userId) {
            return;
        }
        // Make sure only the task owner can delete a task
        if (task.owner !== this.userId) {
            return;
        }
        Tasks.remove(taskId);
    },

    'tasks.setChecked'(taskId, setChecked) {

        check(taskId, String);
        check(setChecked, Boolean);

        const task = Tasks.findOne(taskId);

        // if there is no logged in user then return
        if (! this.userId) {
            return;
        }
        // Make sure only the task owner can check this as done
        if (task.owner !== this.userId) {
            return;
        }
        Tasks.update(taskId, { $set: { checked: setChecked } });
    },

    'tasks.setPrivate'(taskId, setToPrivate) {

        check(taskId, String);
        check(setToPrivate, Boolean);

        const task = Tasks.findOne(taskId);
        // Make sure only the task owner can make a task private
        if (task.owner !== this.userId) {
            throw new Meteor.Error('not-authorized');
        }
        Tasks.update(taskId, { $set: { private: setToPrivate } });
    },

});
```

tasks.test.js

```
/* eslint-env mocha */

import { Meteor } from 'meteor/meteor';
import { Random } from 'meteor/random';
import { assert } from 'chai';
import { Tasks } from './tasks.js';

if (Meteor.isServer) {
    describe('Tasks', () => {
        describe('methods', () => {
            const userId = Random.id();
            let taskId;

            beforeEach(() => {
                Tasks.remove({});
                taskId = Tasks.insert({
                    text: 'test task',
                    createdAt: new Date(),
                    owner: userId,
                    username: 'tmeasday',
                });
            });

            it('can delete owned task', () => {
                // Find the internal implementation of the task method so we can
                // test it in isolation
                const deleteTask = Meteor.server.method_handlers['tasks.remove'];

                // Set up a fake method invocation that looks like what the method
expects
                const invocation = { userId };

                // Run the method with `this` set to the fake invocation
                deleteTask.apply(invocation, [taskId]);

                // Verify that the method does what we expected
                assert.equal(Tasks.find().count(), 0);
            });
        });
    });
}
```

accounts-config.js

```
import { Accounts } from 'meteor/accounts-base';

Accounts.ui.config({
    passwordSignupFields: 'USERNAME_ONLY'
});
```

AccountsUIWrapper.js

```
import React, { Component } from 'react';
import ReactDOM from 'react-dom';
import { Template } from 'meteor/templating';
import { Blaze } from 'meteor/blaze';



export default class AccountsUIWrapper extends Component {

    componentDidMount() {
        // Use Meteor Blaze to render login buttons
        this.view = Blaze.render(Template.loginButtons,
            ReactDOM.findDOMNode(this.refs.container));
    }

    componentWillUnmount() {
        // Clean up Blaze view
        Blaze.remove(this.view);
    }

    render() {
        // Just render a placeholder container that will be filled in
        return <span ref="container" />;
    }
}
```

App.js

```
import React, { Component } from 'react';
import ReactDOM from 'react-dom';
import { withTracker } from 'meteor/react-meteor-data';
import { Tasks } from '../api/tasks.js';
import Task from './Task.js';
import AccountsUIWrapper from './AccountsUIWrapper.js';
// App component - represents the whole app

class App extends Component {

    constructor(props) {
        super(props);
        this.state = {
            hideCompleted: false,
        };
    }

    handleSubmit(event) {
        console.info("Called......");
        event.preventDefault();
        // Find the text field via the React ref
        const text = ReactDOM.findDOMNode(this.refs.textInput).value.trim();
        Meteor.call('tasks.insert', text);
        // Tasks.insert({
        //     text,
        //     createdAt: new Date(), // current time
        //     owner: Meteor.userId(),           // _id of logged in user
        //     username: Meteor.user().username,  // username of logged in user
        // });
        // Clear form
        ReactDOM.findDOMNode(this.refs.textInput).value = '';
    }

    toggleHideCompleted() {
        this.setState({
            hideCompleted: !this.state.hideCompleted,
        });
    }

    renderTasks() {
        let filteredTasks = this.props.tasks;
        if (this.state.hideCompleted) {
            filteredTasks = filteredTasks.filter(task => !task.checked);
        }
        return filteredTasks.map((task) => {
            const currentUserId = this.props.currentUser &&
this.props.currentUser._id;
            const showPrivateButton = task.owner === currentUserId;
            return (
                <Task
                    key={task._id}
                    task={task}
                    showPrivateButton={showPrivateButton}
                />
            );
        });
    }

    render() {
        return (
            <div className="container">
                <header>
                    <h1>Todo List ({this.props.incompleteCount})</h1>

                    <label className="hide-completed">
                        <input
                            type="checkbox"
                            readOnly
                            checked={this.state.hideCompleted}
                            onClick={this.toggleHideCompleted.bind(this)}
                        />
                        Hide Completed Tasks
                    </label>

                    <AccountsUIWrapper />
                    {this.props.currentUser ?
                        <form className="new-task"
onSubmit={this.handleSubmit.bind(this)}>
                            <input
                                type="text"
                                ref="textInput"
                                placeholder="Type to add new tasks"
                            />
                        </form> : ''
                    }
                </header>
                <ul>
                    {this.renderTasks()}
                </ul>
            </div>
        );
    }
}
export default withTracker(() => {
    Meteor.subscribe('tasks');
    return {
        tasks: Tasks.find({}, { sort: { createdAt: -1 } }).fetch(),
        incompleteCount: Tasks.find({ checked: { $ne: true } }).count(),
        currentUser: Meteor.user(),
    };
})(App);
```

Tasks.js

```
import React, { Component } from 'react';
// Task component - represents a single todo item
import { Tasks } from '../api/tasks.js';
import classnames from 'classnames';


export default class Task extends Component {

    toggleChecked() {
        // Set the checked property to the opposite of its current value
        // Tasks.update(this.props.task._id, {
        //     $set: { checked: !this.props.task.checked },
        // });
        Meteor.call('tasks.setChecked', this.props.task._id,
!this.props.task.checked);
    }

    deleteThisTask() {
        //Tasks.remove(this.props.task._id);
        Meteor.call('tasks.remove', this.props.task._id);
    }

    togglePrivate() {
        Meteor.call('tasks.setPrivate', this.props.task._id, !
this.props.task.private);
    }

    render() {
        // Give tasks a different className when they are checked off,
        // so that we can style them nicely in CSS
        const taskClassName = classnames({
            checked: this.props.task.checked,
            private: this.props.task.private,
        });

        return (
            <li className={taskClassName}>
                <button className="delete" onClick={this.deleteThisTask.bind(this)}>
                    &times;
                </button>
                <input
                    type="checkbox"
                    readOnly
                    checked={!!this.props.task.checked}
                    onClick={this.toggleChecked.bind(this)}
                />

                { this.props.showPrivateButton ? (
                    <button className="toggle-private"
onClick={this.togglePrivate.bind(this)}>
                        { this.props.task.private ? 'Private' : 'Public' }
                    </button>
                ) : ''}

                <span className="text">
                    <strong>{this.props.task.username}</strong> :
{this.props.task.text}
                </span>
            </li>
        );
    }
}
```

server/main.js

```
import { Meteor } from 'meteor/meteor';
import '../imports/api/tasks.js';

Meteor.startup(() => {
  // code to run on server at startup
});
```

tests/main.js

```
import assert from "assert";
import "../imports/api/tasks.tests.js";

describe("simple-todos-react", function () {
  it("package.json has correct name", async function () {
    const { name } = await import("../package.json");
    assert.strictEqual(name, "simple-todos");
  });

  if (Meteor.isClient) {
    it("client is not server", function () {
      assert.strictEqual(Meteor.isServer, false);
    });
  }

  if (Meteor.isServer) {
    it("server is not client", function () {
      assert.strictEqual(Meteor.isClient, false);
    });
  }

});
```

package.json, using external mongo from docker container

```
{
  "name": "simple-todos",
  "private": true,
  "scripts": {
    "start": "MONGO_URL=mongodb://localhost:27017/meteor meteor --port 3001 run",
    "test": "meteor test --once --driver-package meteortesting:mocha",
    "test-app": "TEST_WATCH=1 meteor test --full-app --driver-package
meteortesting:mocha",
    "visualize": "meteor --production --extra-packages bundle-visualizer"
  },
  "dependencies": {
    "@babel/runtime": "^7.7.6",
    "bcrypt": "^3.0.8",
    "classnames": "^2.2.6",
    "jquery": "^3.4.1",
    "meteor-node-stubs": "^1.0.0",
    "react": "^16.12.0",
    "react-dom": "^16.12.0"
  },
  "meteor": {
    "mainModule": {
      "client": "client/main.js",
      "server": "server/main.js"ß
    },
    "testModule": "tests/main.js"
  },
  "devDependencies": {
    "chai": "^4.2.0"
  }
}
```

Start_ToDos_Client.sh

```bash
#!/bin/bash

echo --------------------------------------------------
echo --------------------------------------------------
echo -----------------SIMPLE TODOs---------------------
echo --------------------------------------------------
echo --------------------------------------------------
echo
echo
echo Starting Simple ToDos Tutorial....................
echo cd MOD-5-Project/Tutorials/simple-todos...........
# Chnage dir to exec dir
cd MOD-5-Project/Tutorials/simple-todos || exit
pwd
echo
echo
echo Running npm start............................
# You can set the connection string for any mongo instance here
# in the package.json file on the npm start line
# defualt for meteor is embedded mongo.
# start with npm to use selected Mongo running, (e.g in Docker or lan instance)
# in my case I am running mongo in a Docker container
npm start
echo Stopping Simple ToDos........................
echo
echo
echo --------------------------------------------------
echo --------------------------------------------------
echo ---------SHUTDOWN SIMPLE TODOs COMPLETE-----------
echo --------------------------------------------------
echo --------------------------------------------------
```