**Exercise06\_03\_01 – Step 1**

navbar.html

<nav class="navbar navbar-static-top navbar-inverse">

<div class="container-fluid">

<div class="navbar-header">

<a class="navbar-brand"   
 href="https://github.com/Swiip/generator-gulp-  
 angular">

<span class="glyphicon glyphicon-home"></span>   
 My Message Board

</a>

</div>

<div class="collapse navbar-collapse" id="bs-example-  
 navbar-collapse-6">

<ul class="nav navbar-nav">

<li class="active"><a ng-href="#">Home</a></li>

<li><a ng-href="#">Login</a></li>

</ul>

</div>

</div>

</nav>

------------------------------------

auth.html

hello

------------------------------------

auth.controller.js

export class AuthController {

}

------------------------------------

index.route.js

})

.state('auth', {

url: '/auth',

templateUrl: 'app/auth/auth.html',

controller: 'AuthController',

controllerAs: 'auth'

});

index.module.js

import { MainController } from './main/main.controller';

import { AuthController } from './auth/auth.controller';

------------------------------------

index.module.js

.controller('MainController', MainController)

.controller('AuthController', AuthController)

------------------------------------

index.html

<body>

<div>

<acme-navbar></acme-navbar>

</div>

------------------------------------

navbar.html

<li><a ng-href="#/auth">Login</a></li>

------------------------------------

navbar.html

<li><a ui-sref="auth">Login</a></li>

------------------------------------

auth.html

<div class="container">

<div class="panel panel-default">

<div class="panel-heading">Post A Message</div>

<div class="panel-body">

</div>

</div>

</div>

auth.html

<div class="container">

<div class="col-md-6">

<div class="panel panel-default">

<div class="panel-heading">Post A Message</div>

<div class="panel-body">

</div>

</div>

</div>

<div class="col-md-6">

<div class="panel panel-default">

<div class="panel-heading">Post A Message</div>

<div class="panel-body">

</div>

</div>

</div>

<</div>

------------------------------------

auth.html

<div class="panel-heading">Login</div>

<div class="panel-body">

<form>

<div class="form-group">

<label>Email address</label>

<input type="email"   
 class="form-control">

</div>

</form>

</div>

</div>

------------------------------------

auth.html

<div class="form-group">

<label>Email address</label>

<input type="email" class="form-  
 control">

</div>

<div class="form-group">

<label>Password</label>

<input type="password" class="form-  
 control">

</div>

auth.html

<button type="submit"   
 class="btn btn-default">Submit</button>

</form>

------------------------------------

server.js

<div class="panel-heading">Register</div>

<div class="panel-body">

<form>

<div class="form-group">

<label>Email address</label>

<input type="email"   
 class="form-control">

</div>

<div class="form-group">

<label>Password</label>

<input type="password"   
 class="form-control">

</div>

<div class="form-group">

<label>Password Confirm</label>

<input type="password"   
 class="form-control">

</div>

<button type="submit"   
 class="btn btn-default">Submit</button>

</form>

</div>

**Exercise06\_03\_01 – Step 2**

compareTo.directive.js

export function CompareToDirective($parse) {

'ngInject'

}

------------------------------------

index.module.js

import { AuthController } from './auth/auth.controller';

import { CompareToDirective } from   
 './directives/compareTo.directive';

------------------------------------

index.module.js

.directive('acmeMalarkey', MalarkeyDirective)

.directive('compareTo', CompareToDirective);

------------------------------------

auth.html

<form name="register">

------------------------------------

auth.html

<label>Password</label>

<input type="password"   
 class="form-control" name=”pwd”

ng-model="pwd">

------------------------------------

auth.html

<label>Password Confirm</label>

<input type="password"   
 class="form-control" name=”pwdConfirm”

ng-model="pwdConfirm">

------------------------------------

auth.html

<label>Password Confirm</label>

<input type="password"   
 class="form-control"

compare-to="pwd"

ng-model="pwdConfirm">

compareTo.directive.js

'ngInject'

return {

require: 'ngModel',

link: function(scope, elm, attrs, ngModel) {

}

}

------------------------------------

compareTo.directive.js

link: function (scope, elm, attrs, ngModel) {

var mainModel = $parse(attrs.compareTo);

var secondModel = $parse(attrs.ngModel);

}

------------------------------------

compareTo.directive.js

var secondModel = $parse(attrs.ngModel);

scope.$watch(attrs.ngModel, function(newValue) {

ngModel.$setValidity(attrs.name,   
 newValue === mainModel(scope));

});

------------------------------------

compareTo.directive.js

scope.$watch(attrs.ngModel, function(newValue) {

ngModel.$setValidity(attrs.name,   
 newValue === mainModel(scope));

});

scope.$watch(attrs.compareTo, function(newValue) {

ngModel.$setValidity(attrs.name,   
 newValue === secondModel(scope));

});

------------------------------------

auth.html

<span   
 ng-show="register.pwdConfirm.$invalid">

Passwords do not match.

</span>

<button type="submit"   
 class="btn btn-default">Submit</button>

auth.html

<span ng-show="register.pwdConfirm.$dirty &&   
 register.pwdConfirm.$invalid">

**Exercise06\_03\_01 – Step 3**

front-end terminal

bower install satellizer --save

------------------------------------

index.module.js

angular.module('myMessageFront', ['ui.router', 'ui.bootstrap', 'toastr', 'satellizer'])

------------------------------------

index.config.js

export function config ($logProvider,   
 toastrConfig, $authProvider) {

'ngInject';

------------------------------------

index.config.js

toastrConfig.progressBar = true;

$authProvider.signupUrl = '';

------------------------------------

index.module.js

.constant('API\_URL', 'http://localhost:8080/')

.constant('malarkey', malarkey)

------------------------------------

index.config.js

export function config($logProvider, toastrConfig, $authProvider, API\_URL) {

'ngInject';

------------------------------------

index.config.js

$authProvider.signupUrl = API\_URL + 'auth/register';

------------------------------------

auth.html

<form name="register"

ng-submit="auth.register()">

auth.controller.js

export class AuthController {

register() {

console.log('test');

}

}

------------------------------------

auth.controller.js

export class AuthController {

constructor($auth) {

'ngInject';

this.$auth = $auth;

}

------------------------------------

auth.controller.js

register() {

this.$auth.signup({email: 'test@test.com'});

}

**Exercise06\_03\_01 – Step 4**

server.js

app.post('/auth/register', (req, res) => {

console.log(req.body);

});

------------------------------------

auth.html

<label>Email address</label>

<input type="email" class="form-control"

ng-model="auth.user.email">

------------------------------------

auth.html

<label>Password</label>

<input type="password"

class="form-control" name="pwd"

ng-model="auth.user.pwd">

------------------------------------

auth.html

<label>Password Confirm</label>

<input type="password"

class="form-control" name="pwdConfirm"

compare-to="auth.user.pwd"

ng-model="pwdConfirm">

------------------------------------

auth.controller.js

register() {

this.$auth.signup(this.user);

}

------------------------------------

auth.html

<button type="submit"

class="btn btn-default"

ng-disabled="register.pwdConfirm.$invalid">

Submit</button>

**Exercise06\_03\_01 – Step 5**

Message.js

var mongoose = require('mongoose');

module.exports = mongoose.model('Message', {

msg: String

});

------------------------------------

server.js

var url = 'mongodb://localhost:27017/' + dbName;

var Message = require('./models/message');

------------------------------------

User.js

module.exports = mongoose.model('User', {

email: String,

pwd: String

});

------------------------------------

server.js

var Message = require('./models/message');

var User = require('./models/user');

------------------------------------

server.js

app.post('/auth/register', (req, res) => {

console.log(req.body);

var user = new User(req.body);

user.save((err, result) => {

if (err) {

res.status(500).send({

message: err.message

});

}

else {

res.status(200);

}

});

});

auth.js

module.exports = {

register: () => {

}

}

------------------------------------

auth.js

register: (req, res) => {

console.log(req.body);

var user = new User(req.body);

user.save((err, result) => {

if (err) {

res.status(500).send({

message: err.message

});

}

res.status(200);

});

}

------------------------------------

server.js

app.post('/auth/register', auth.register);

------------------------------------

server.js

var User = require('./models/user');

var auth = require('./controllers/auth');

------------------------------------

auth.js

var User = require('../models/user');

module.exports = {

------------------------------------

message.js

module.exports = {

get:

post:

}

message.js

var Message = require('../models/message');

module.exports = {

get: (req, res) => {

Message.find({}).exec((err, result) => {

res.send(result);

})

},

post: (req, res) => {

console.log(req.body);

var message = new Message(req.body);

message.save();

res.status(200);

}

}

------------------------------------

server.js

var auth = require('./controllers/auth');

var message = require('./controllers/message');

------------------------------------

auth.js

module.exports = {

register: (req, res) => {

console.log(req.body);

User.findOne({

email: req.body.email

}, (err, existingUser) => {

var user = new User(req.body);

user.save((err, result) => {

if (err) {

return res.status(500).send({

message: err.message

});

}

res.status(200);

});

});

}

}

auth.js

}, (err, existingUser) => {

if (existingUser) {

return res.status(409).send({

message: 'Email is already registered'

});

}

------------------------------------

back-end terminal

npm install jwt-simple -save

------------------------------------

auth.js

var User = require('../models/user');

var jwt = require('jwt-simple');

------------------------------------

auth.js

function createToken(user) {

}

------------------------------------

back-end terminal

npm install moment -save

------------------------------------

auth.js

var jwt = require('jwt-simple');

var moment = require('moment');

------------------------------------

auth.js

function createToken(user) {

var payload = {

sub: user.\_id,

iat: moment().unix(),

exp: moment().add(14, 'days').unix()

}

}

auth.js

exp: moment().add(14, 'days').unix()

}

return jwt.encode(payload, 'secret');

------------------------------------

auth.js

}

res.status(200).send({

token: createToken(result)

});

------------------------------------

auth.controller.js

register() {

var vm = this;

this.$auth.signup(this.user)

.then(function(token) {

vm.$auth.setToken(token);

});

}

**Exercise06\_03\_01 – Step 6**

server.js

function checkAuthenticated(req, res, next) {

}

------------------------------------

server.js

function checkAuthenticated(req, res, next) {

if (!req.header('Authorization')) {

return res.status(401).send({

message: 'Please make sure your request has an   
 authorization header'

});

}

}

------------------------------------

server.js

var message = require('./controllers/message');

var jwt = require(('jwt-simple');)

------------------------------------

server.js

app.post('/api/message', checkAuthenticated, auth.register);

------------------------------------

server.js

if (!req.header('Authorization')) {

return res.status(401).send({

message: 'Please make sure your request has an   
 authorization header'

});

}

var token = req.header('Authorization').split(' ')[1];

console.log(token);

next();

}

server.js

var token = req.header('Authorization').split(' ')[1];

console.log(token);

var payload = jwt.decode(token, 'secret');

console.log(payload);

next();

------------------------------------

server.js

var jwt = require('jwt-simple');

var moment = require('moment');

------------------------------------

server.js

var payload = jwt.decode(token, 'secret');

if (payload.exp <= moment().unix()) {

return res.status(401).send({

message: 'Token has expired'

});

}

next();

------------------------------------

server.js

var token = req.header('Authorization').split(' ')[1];

var payload = jwt.decode(token, 'secret');

if (payload.exp <= moment().unix()) {

return res.status(401).send({

message: 'Token has expired'

});

}

req.user = payload.sub;

next();

------------------------------------

checkAuthenticated.js

var jwt = require('jwt-simple');

var moment = require('moment');

module.exports = function checkAuthenticated(req, res, next) {

server.js

var message = require('./controllers/message');

var checkAuthenticated =   
 require('./services/checkAuthenticated');

------------------------------------

cors.js

module.exports = (req, res, next) => {

res.header('Access-Control-Allow-Origin', '\*');

res.header('Access-Control-Allow-Headers', 'Content-Type,   
 Authorization');

next();

}

------------------------------------

server.js

var checkAuthenticated =   
 require('./services/checkAuthenticated');

var cors = require('./services/cors');

------------------------------------

server.js

app.use(bodyParser.json());

app.use(cors);

------------------------------------

controllers/message.js

post: (req, res) => {

console.log(req.body, req.user);

------------------------------------

models/Message.js

module.exports = mongoose.model('Message', {

msg: String,

user: {type: mongoose.Schema.ObjectId, ref: 'User'}

});

------------------------------------

controllers/message.js

post: (req, res) => {

console.log(req.body, req.user);

req.body.user = req.user;

controllers/message.js

get: (req, res) => {

Message.find({}).populate('user', '-pwd').exec((err,   
 result) => {

res.send(result);

})

},

------------------------------------

main.html

<li class="list-group-item"

ng-repeat="message in main.messages">

{{message.msg}} {{message.user.email}}

</li>