

Simple MEAN Auth

User authentication is sometime a very difficult subject. What am I saying, it is always a bear to get working correctly. However, in this sample tutorial, we are going to try a very simple user authentication using the full mean stack.

Server side:

Routes:

We are going to dissect this code bit by bit. There is a great deal going on, but we will see why each piece is needed.

```
var express = require('express');
var router = express.Router();
var passport = require('passport');

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

router.post('/register', function(req, res) {
  User.register(new User({ username: req.body.username }),
    req.body.password, function(err, account) {
    if (err) {
      return res.status(500).json({
        err: err
      });
    }
    passport.authenticate('local')(req, res, function () {
      return res.status(200).json({
        status: 'Registration successful!'
      });
    });
  });
});

router.post('/login', function(req, res, next) {
  passport.authenticate('local', function(err, user, info) {
    if (err) {
      return next(err);
    }
  });
});
```

```

    }
    if (!user) {
      return res.status(401).json({
        err: info
      });
    }
    req.logIn(user, function(err) {
      if (err) {
        return res.status(500).json({
          err: 'Could not log in user'
        });
      }
      res.status(200).json({
        status: 'Login successful!'
      });
    });
  })(req, res, next);
});

router.get('/logout', function(req, res) {
  req.logout();
  res.status(200).json({
    status: 'Bye!'
  });
});

router.get('/status', function(req, res) {
  if (!req.isAuthenticated()) {
    return res.status(200).json({
      status: false
    });
  }
  res.status(200).json({
    status: true
  });
});

module.exports = router;

```

Angular side:

Main Application:

Router

```
$routeProvider
.when('/', {
  templateUrl: 'html/views/home.html',
  access: {restricted: true}
})
.when('/login', {
  templateUrl: 'html/views/login.html',
  controller: 'LoginController',
  access: {restricted: false}
})
.when('/logout', {
  controller: 'LogoutController',
  access: {restricted: true}
})
.when('/register', {
  templateUrl: 'html/views/register.html',
  controller: 'RegisterController',
  access: {restricted: false}
})
.otherwise({
  redirectTo: '/'
});
```

Run Function

```
$rootScope.$on('$routeChangeStart',
function (event, next, current) {
  AuthService.getUserStatus();
  if (next.access.restricted &&
    !AuthService.isLoggedIn()) {
    $location.path('/login');
    $route.reload();
  }
});
```

Controllers:

Login Controller

```
$scope.login = function () {  
  
    // initial values  
    $scope.error = false;  
    $scope.disabled = true;  
  
    // call login from service  
    AuthService.login($scope.loginForm.username, $scope.loginForm.password)  
    // handle success  
    .then(function () {  
        $location.path('/');  
        $scope.disabled = false;  
        $scope.loginForm = {};  
    })  
    // handle error  
    .catch(function () {  
        $scope.error = true;  
        $scope.errorMessage = "Invalid username and/or password";  
        $scope.disabled = false;  
        $scope.loginForm = {};  
    });  
  
};
```

Logout Controller

```
$scope.logout = function () {  
  
    // call logout from service  
    AuthService.logout()  
    .then(function () {  
        $location.path('/login');  
    });  
  
};
```

RegisterController

```

$scope.register = function () {

    // initial values
    $scope.error = false;
    $scope.disabled = true;

    // call register from service
    AuthService.register($scope.registerForm.username, $scope.registerForm.password)
    // handle success
    .then(function () {
        $location.path('/login');
        $scope.disabled = false;
        $scope.registerForm = {};
    })
    // handle error
    .catch(function () {
        $scope.error = true;
        $scope.errorMessage = "Something went wrong!";
        $scope.disabled = false;
        $scope.registerForm = {};
    });

};

```

Authentication Service

Why do you need a service? This is going to take some thinking and thinking...

AuthService

```

// create user variable
var user = null;

// return available functions for use in the controllers
return ({
    isLoggedIn: isLoggedIn,
    getUserStatus: getUserStatus,
    login: login,
    logout: logout,
    register: register
});

function isLoggedIn() {
    if(user) {

```

```

        return true;
    } else {
        return false;
    }
}

function getUserStatus() {
    $http.get('/user/status')
    // handle success
    .success(function (data) {
        if(data.status){
            user = true;
        } else {
            user = false;
        }
    })
    // handle error
    .error(function (data) {
        user = false;
    });
}

function login(username, password) {

    // create a new instance of deferred
    var deferred = $q.defer();

    // send a post request to the server
    $http.post('/user/login',
        {username: username, password: password})
    // handle success
    .success(function (data, status) {
        if(status === 200 && data.status){
            user = true;
            deferred.resolve();
        } else {
            user = false;
            deferred.reject();
        }
    })
    // handle error
    .error(function (data) {
        user = false;
    });
}

```

```

        deferred.reject();
    });

    // return promise object
    return deferred.promise;
}

function logout() {

    // create a new instance of deferred
    var deferred = $q.defer();

    // send a get request to the server
    $http.get('/user/logout')
        // handle success
        .success(function (data) {
            user = false;
            deferred.resolve();
        })
        // handle error
        .error(function (data) {
            user = false;
            deferred.reject();
        });

    // return promise object
    return deferred.promise;
}

function register(username, password) {

    // create a new instance of deferred
    var deferred = $q.defer();

    // send a post request to the server
    $http.post('/user/register',
        {username: username, password: password})
        // handle success
        .success(function (data, status) {
            if(status === 200 && data.status){
                deferred.resolve();
            }
        })
        .error(function (data, status) {
            deferred.reject();
        });

    // return promise object
    return deferred.promise;
}

```

```
    } else {  
      deferred.reject();  
    }  
  })  
  // handle error  
  .error(function (data) {  
    deferred.reject();  
  });  
  
  // return promise object  
  return deferred.promise;  
}
```