Web Tech Coursework 2

Ihtisham Nadeem Chaudhry

40409062

Introduction

On extending my first project and taking it to the next level which is more of a client based website on a real platform. Which consists of real time interface of a user with a website registering and longing into the website in order to use cyphers for encrypting their text. Every user has to first register on the website and than login in order for them to use the cyphers. The website consists of:

1) Register page.
2) Login page.
3) Home page.
4) Ceaser page.
5) Rot 13 page.
6) Vigenere page.

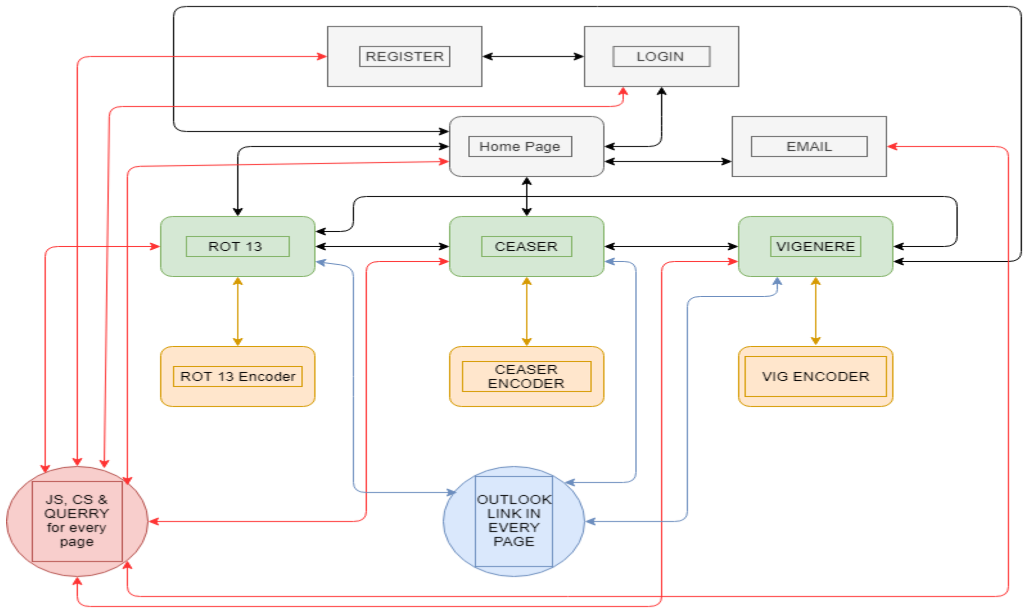
Mongo database has been used for the login and register forms and entire data of user's username and password is being backing up in it and saving into it.

Software Design

For my new project, I needed to make a login and registration forms in which a user can register and login on the web site. On opening the website, if the user is already registered on my website then login form appears if not then the user must register first to login in order to use the cyphers.

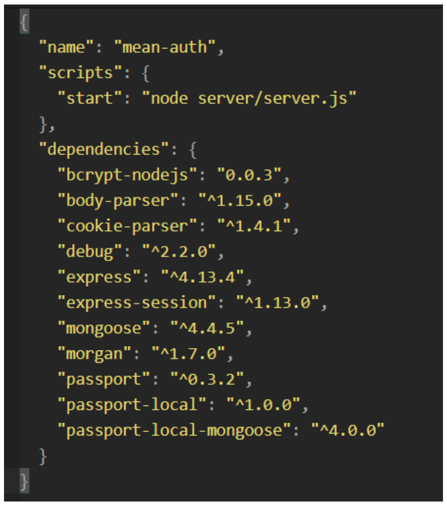
Following are the requirements for the project:

1) Node JS.
2) Mongo database.
3) Passport module.
4) Login form.
5) Registration form.
6) Java scripts for every html page.
7) First project files.



Implementation

After making my software design for my project, I was required to work on Node JS. As I was not familiar with Node JS at the start of my work that gave me some difficulties to begin with my work. However, I took some help by looking at the lab work and lectures there was a lot of information about Node JS and how to use it. I did a lot of internet research for my project and about Node JS. It was found that there are many ways of implementing my project on Node JS.

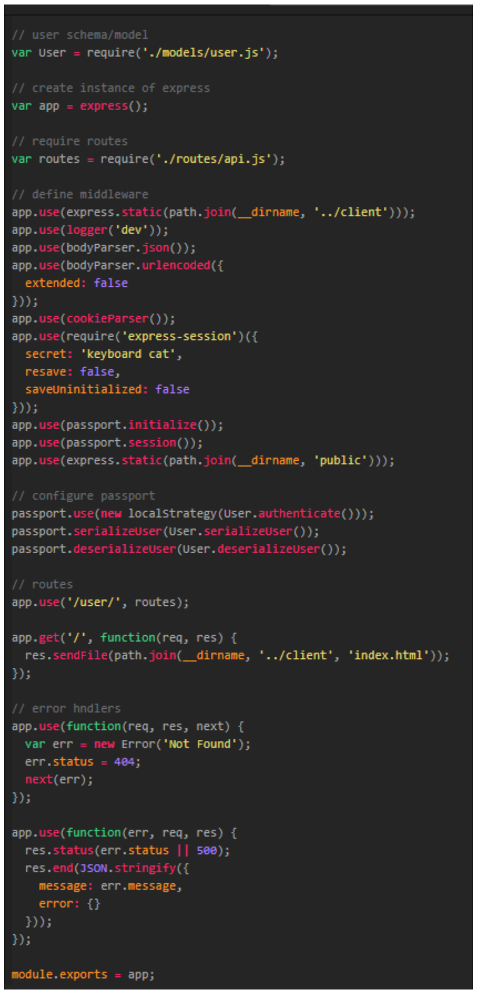


```
{
  "name": "mean-auth",
  "scripts": {
    "start": "node server/server.js"
  },
  "dependencies": {
    "bcrypt-nodejs": "0.0.3",
    "body-parser": "^1.15.0",
    "cookie-parser": "^1.4.1",
    "debug": "^2.2.0",
    "express": "^4.13.4",
    "express-session": "^1.13.0",
    "mongoose": "^4.4.5",
    "morgan": "^1.7.0",
    "passport": "^0.3.2",
    "passport-local": "^1.0.0",
    "passport-local-mongoose": "^4.0.0"
  }
}
```

Every module used in the project has its own purpose for completion of a good website.

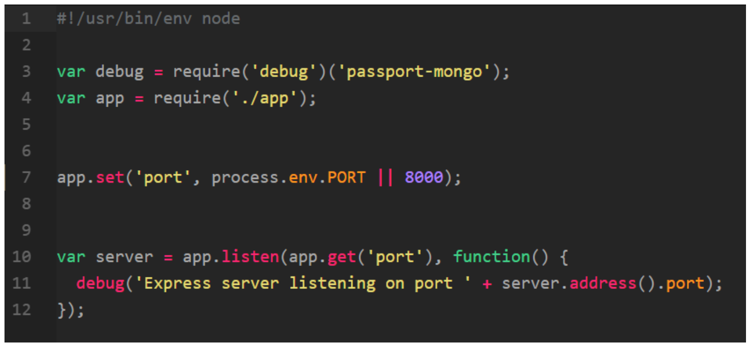
1) Mongoose: mongoose has been used for the mongo DB for storing the data of the users their usernames and passwords.
2) Bcyrypt-Node JS: it has been used for hashing the user password.

3) Body-parser: incoming request from the client end side.
4) Cookie-parser: cookies used for a better streaming of the request from the client.
5) Debug: listening to the ports on which my website and mongo DB has been set if there is any problem it will try to resolve or give and error.
6) Express: module for the web deployment incoming and outgoing connections.
7) Morgan: HTTP requests.
8) Passport: authentication middleware for the Node JS for using passwords and usernames.



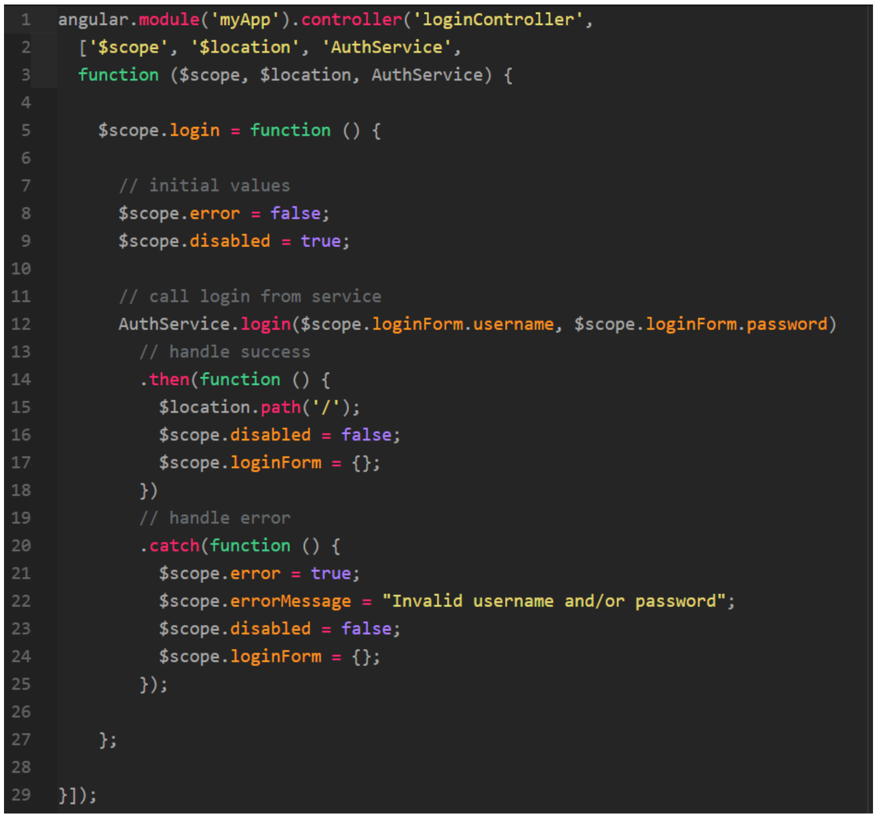
```
// user schema/model
var User = require('./models/user.js');

// create instance of express
var app = express();

// require routes
var routes = require('./routes/api.js');

// define middleware
app.use(express.static(path.join(__dirname, '../client')));
app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({
  extended: false
}));
app.use(cookieParser());
app.use(require('express-session')({
  secret: 'keyboard cat',
  resave: false,
  saveUninitialized: false
}));
app.use(passport.initialize());
app.use(passport.session());
app.use(express.static(path.join(__dirname, 'public')));

// configure passport
passport.use(new localStrategy(User.authenticate()));
passport.serializeUser(User.serializeUser());
passport.deserializeUser(User.deserializeUser());

// routes
app.use('/user/', routes);

app.get('/', function(req, res) {
  res.sendFile(path.join(__dirname, '../client', 'index.html'));
});

// error hndlers
app.use(function(req, res, next) {
  var err = new Error('Not Found');
  err.status = 404;
  next(err);
});

app.use(function(err, req, res) {
  res.status(err.status || 500);
  res.end(JSON.stringify({
    message: err.message,
    error: {}
  }));
});

module.exports = app;
```

After installing my dependencies for my project I need to tell every module how to work in the diagram a code has been written for every module to work for my website.



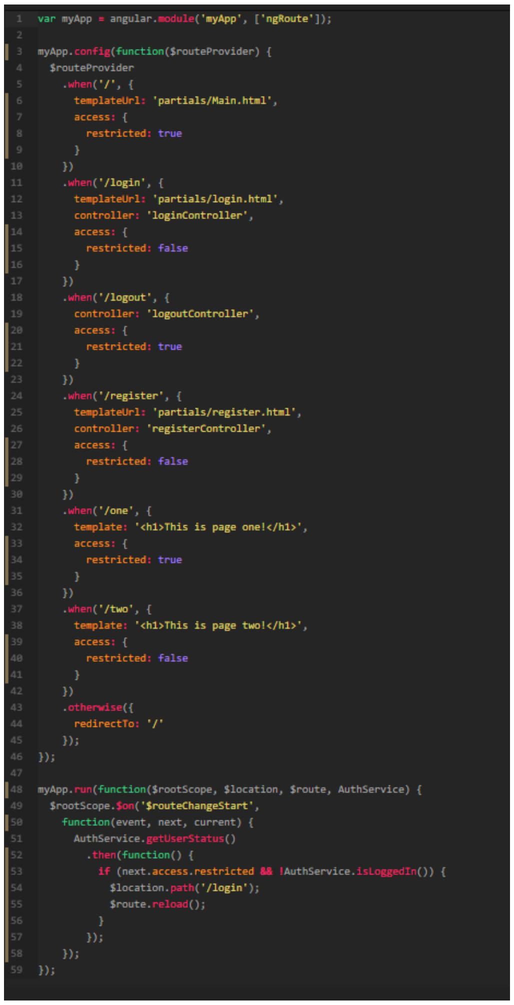
```
#!/usr/bin/env node

var debug = require('debug')('passport-mongo');
var app = require('./app');


app.set('port', process.env.PORT || 8000);


var server = app.listen(app.get('port'), function() {
  debug('Express server listening on port ' + server.address().port);
});
```

Code for my server deployment on a local machine.



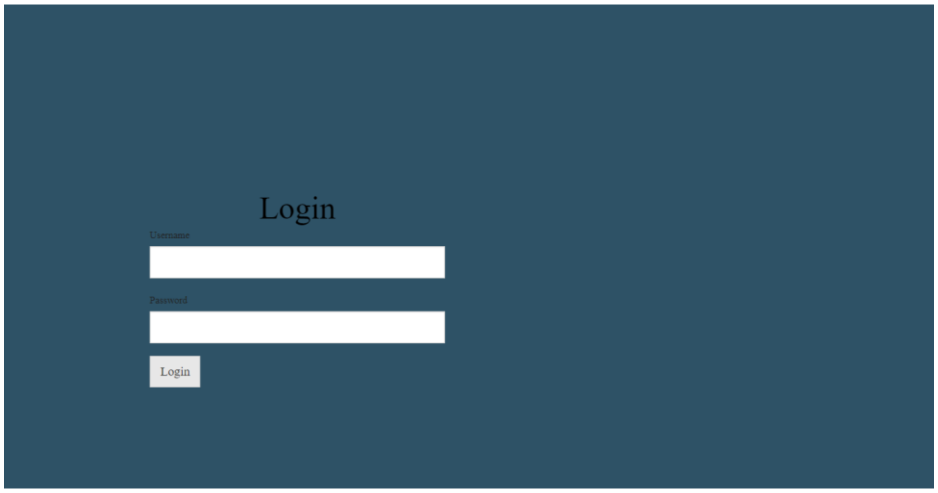
```
angular.module('myApp').controller('loginController',
  ['$scope', '$location', 'AuthService',
  function ($scope, $location, AuthService) {

    $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 = {};
        });

    };

}]);
```

In this diagram there is a code for my login and registration forms, if the user has already registered. On logging the code get execute and checks the username and password if the matches in the database of my project then it redirects it to the home page of my cyphers. If the user name or password is not correct it shows error username or password is not correct to the user.

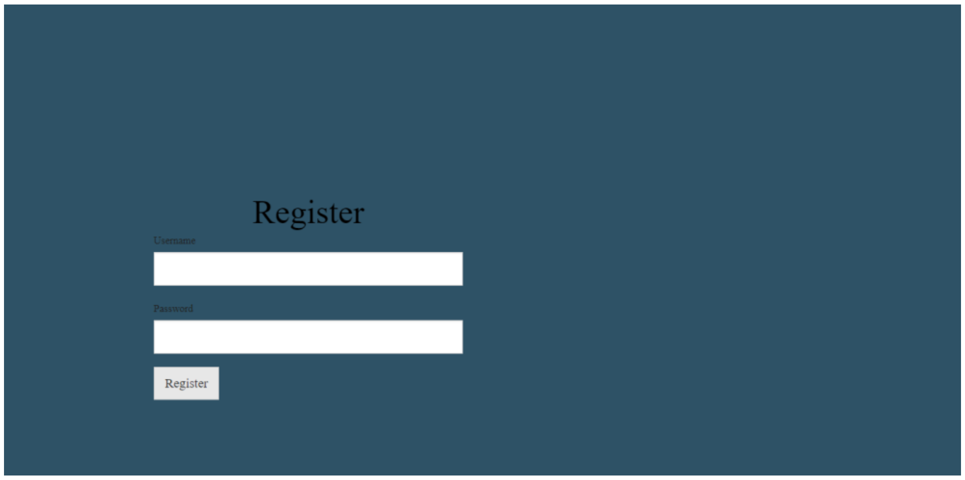


```javascript
var myApp = angular.module('myApp', ['ngRoute']);

myApp.config(function($routeProvider) {
  $routeProvider
    .when('/', {
      templateUrl: 'partials/Main.html',
      access: {
        restricted: true
      }
    })
    .when('/login', {
      templateUrl: 'partials/login.html',
      controller: 'loginController',
      access: {
        restricted: false
      }
    })
    .when('/logout', {
      controller: 'logoutController',
      access: {
        restricted: true
      }
    })
    .when('/register', {
      templateUrl: 'partials/register.html',
      controller: 'registerController',
      access: {
        restricted: false
      }
    })
    .when('/one', {
      template: '<h1>This is page one!</h1>',
      access: {
        restricted: true
      }
    })
    .when('/two', {
      template: '<h1>This is page two!</h1>',
      access: {
        restricted: false
      }
    })
    .otherwise({
      redirectTo: '/'
    });
});

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

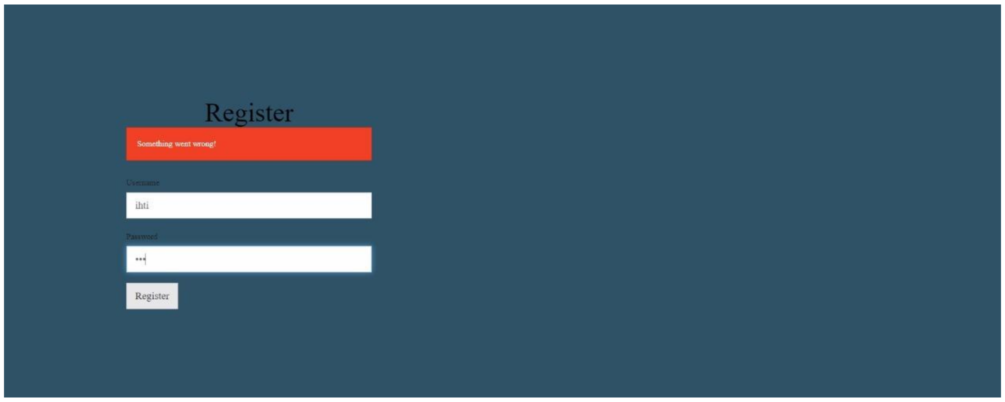
All the code for the routes and for every page using angular module the codes knows how to act on the client requests after the user has logged in, if a user requests for the Rot 13 cypher. This code can get the Rot 13 page and sends it to the user’s screen.



Login page



Register page


Something went wrong!

Username: ihti

Password: •••

Register can get an error if the same username has already been taken by someone else who has registered before a new user can register for himself. A new registration of a username always has to be a unique which is different from others.

Critical Evaluation

Requirement for this project was to make a login and registration for client end system. After that merging the first project with the new project, now a user can use the cyphers once he has registered and logged in.

I did not read carefully the descriptor for our coursework 2. I have missed out the main part which was Node JS platform. I did most of my work on WAMPP but before writing the report I had a talk on project with one on my classmates and it came into my knowledge that coursework has to be done on Node JS platform only. I did not had enough time as the extended deadline was almost come to an end and I started working on Node JS by looking on the lab notes as well as internet reading and looking some similar type of projects it got me an idea how to work with Node JS.

In the beginning, I tried to implement my WAMMP work on Node JS but I faced many issues on my functions which were unable to call on the local host. Trying different ways, I tired with jade module but again that gave me many issues. However, I tried with angular module and somehow that get me going and most of my functions were working quite well and there were few issues which I managed to solve with the help of Stack overflow website and guidance from a professor back in Dubai.

On using mongo database as my machine has windows not a server windows installed I faced many issues when connecting my modules to the database. I resolved every issue by searching the problem over the internet.

I missed out the email part which has to be there after the login page but I have included outlook web address with an external link on the home page, if the user wants to send encrypted text to any other friend or users. As the time for the late submission is almost near to end the email part will be available in the coursework.

Website can be improved in many ways, a page for the users account a user can update his email address, phone number, display name etc. A method can be introduced where a user can send the encrypted text to the users who has registered on the website. Another method where a user can login into the website by using their Facebook account details or using Gmail ID. While registering on the website, a verification email can be send to their private email address for verification the user is genuine and for security purposes.

As the code for my first attempt is long. I will be uploading on GitHub along with my Coursework.

Personal Evaluation

The demonstration for the project seems easy. But I did a mistake by not working on the Node JS platform in the beginning. I have missed out the whole point of this module which was working on Node JS platform. My project was almost complete on the WAMPP server but I shifted to Node JS and it gets complicated when a project is transferred to another Platform like in my case I did. By facing many difficulties only 25% of my code were working on Node JS but that was not enough I had to make full 100% in order for me to get the website working correctly. I learned a lot of stuff for Node JS from the internet by looking into many tutorials and reading posts of beginner for Node JS. It was a bit difficult for me to complete the coursework with short amount of time & I learned many things but couldn’t implemented into my project because of the time. Node JS is a good platform but making a small mistake in any part of the code can make the whole website stop working properly. The good part is that few modules which were used like debug shows us the problem where the problem lies and we can amend our code. Most of the errors were resolved by looking for the answers on the forms like stack overflow and GitHub.

In my opinion I have not performed so well it could have been better than this. If I had not had misunderstanding on using the right platform.