

Lab Assignment 3

Mean Stack

Contributors

Team : 4-2

Member 1

- Name : Rahul Reddy Yerva
- Class id : 59

Member 2

- Name : Kavın Kumar Arumugam
- Class id : 63

link : <https://ase5551-lab-3.herokuapp.com/>

Objective

1. [To Create a database in MongoDB and store data related to the application](#)
2. [To implement CRUD operations by using REST services on MongoDB related to the application](#)
3. [Add a node module related to the application](#)
4. [Design good user interface](#)




Design and Screenshots/Code Snapshots

1. Database connection and login, register activity

We have used [mlab](#) for database connection to store user details and performed login action and register action. We have also validated the actions and made alert messages from the server side.

mlab collection

records / page 10 ▾ [1 - 4 of 4]

_id	user_name	password	email_id	
5bda6756de04d11dcc937e37	kavin	12345	kavin@gmail.com	
5bda67e1de04d11dcc937e38	rahul	12345	rahul@gmail.com	
5bda71dad2aa23000465c63f	jack	123456	jack@gmail.com	
5bda78a7d2aa23000465c640	andrew	123456	andrew@gmail.com	

records / page 10 ▾ [1 - 4 of 4]

mlab connection code

```
const database = "mongodb://kaphc:Xyloa7707@ds147723.mlab.com:47723/lab-3";

mongoose.Promise = global.Promise;
mongoose.connect(database, function (err) {
  if(err){
    console.error("Error : " + err);
  }
});
```

Login Page

Welcome to Group Chat

Username

Password

☒ Remember me

Login

or

Need an Account? [click here](#)

Login Page with Validation

ase5551-lab-3.herokuapp.com says
Invalid credentials

OK

Username

john

Password

....

☒ Remember me

Login

or

Need an Account? [click here](#)

Signup Page

Sign Up

First Name

Password

Email Address

☐ I agree with the [Terms and Conditions](#).

Register

Signup Page with Validation

Sign Up

First Name

andrew

Password

123456

Email Address

andrew@

☒ I agree with the [Terms and Conditions](#).

Register

2. To perform CRUD operations

Call from client

```
app.controller('login_controller', function ($scope, $http) {

    $scope.login = function () {
        let url_req = 'lab-3/login?' +
            'user_name=' + $scope.formData.user_name + '&' +
            'password=' + $scope.formData.password + '&';
        var req = $http.post(url_req, $scope.formData).then(function (response) {
            if(response.data['success']){
                localStorage['logged_user'] = $scope.formData.user_name;
                location.href = "chat.html";
            }
            else{
                alert("Invalid credentials");
            }
        });
    };

    $scope.register = function () {
        let url_req = 'lab-3/register?' +
            'user_name=' + $scope.formData.user_name + '&' +
            'password=' + $scope.formData.password + '&' +
            'email_id=' + $scope.formData.email_id + '&';
        var req = $http.post(url_req, $scope.formData).then(function (response) {
            if(response.data['success']){
                location.href = "login.html";
                alert("Registered successfully");
            }
            else{
                alert("Error while registering");
            }
        });
    };
});
```

Routes Code

```
router.post('/login', function(req, res){
  console.log('login a user');
  const user_name = req.body.user_name;
  const password = req.body.password;

  user.authenticate(user_name, password, (err, user) => {
    if (err) throw err;
    if (!user) {
      return res.json({success: false, msg: "User not found"});
    }
    console.log("logged in");
    return res.json({success: true, msg: "User found"});
  });
});

router.post('/register', function(req, res){
  var newUser = new user();

  newUser.user_name = req.body.user_name;
  newUser.password = req.body.password;
  newUser.email_id = req.body.email_id;

  newUser.save(function(err, insertUser){
    if(err){
      console.log("Error : " + err);
      res.json({success: false, msg: "failed to register"})
    } else {
      res.json({success: true, msg: "Successfully Registered!!!!"});
    }
  });
});
```

Server Code

```
// required packages
const express = require('express');
const bodyParser = require('body-parser');
const path = require('path');
const socket = require('socket.io');

// app set up
var app = express();
const port = process.env.PORT || 3000;
var server = app.listen(port, function(){
  console.log("server started at port number : " + port);
});
app.use(bodyParser.urlencoded({extended: true}));
app.use(bodyParser.json());
const route = require('./routes/users');
app.use('/lab-3', route);

// including static files
app.use(express.static(path.join(__dirname, 'public')));

app.get('*', (req, res) => {
  res.sendFile(path.join(__dirname, 'public/index.html'));
});
```


3. Add a node module

We have used socket.io node module to the application where users can have a group chat among them.

Socket.io implementation code

```
// socket connection
var socket = io.connect('');

var message = document.getElementById('message');
var handle = localStorage['logged_user'] || 'Your friend';
var button = document.getElementById('send');
var output = document.getElementById('output');
var feedback = document.getElementById('feedback');

button.addEventListener('click', function(){
    socket.emit('chat',{
        message: message.value,
        handle: handle
    });
    message.value = "";
});

message.addEventListener('keypress', function(){
    socket.emit('typing', handle)
});

socket.on('chat',function (data) {
    feedback.innerHTML = '';
    output.innerHTML += '<p><strong>' + data.handle + ':</strong>' + data.message + '</p>';
});

socket.on('typing',function (data) {
    feedback.innerHTML = '<p><em>' + data + ' is typing a message....' + '</em></p>';
});
```


We store logged user detail in local storage

The screenshot shows the Chrome DevTools Application panel. The left sidebar has three sections: Application, Storage, and Cache. Under Application, there are icons for Manifest, Service Workers, and Clear storage. Under Storage, there is a dropdown for Local Storage, which is expanded to show a list of origins. The origin <https://ase551-lab-3.herokuapp.com/> is selected and highlighted in blue. Under Cache, there are icons for Cache Storage and Application Cache. The main panel displays a table of Local Storage data for the selected origin. The table has two columns: Key and Value. The first row shows the key 'logged_user' with the value 'kavin'. Below the table, there is a text input field containing 'kavin' and a 'Line 1, Column 1' status bar at the bottom.

Key	Value
logged_user	kavin

1 kavin

Line 1, Column 1

Chat box

Message

Send

Chat box with typing message

kavin:hello

kavin:hey, I'm testing out group messaging.

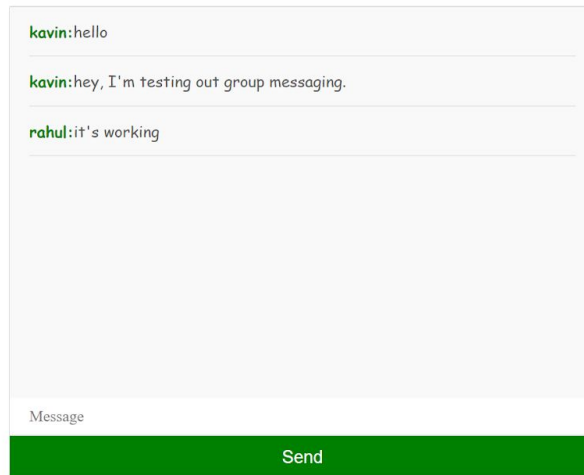
rahul:it's working

rahul is typing a message....

Message

Send

Chat box with group message



A screenshot of a chat box interface. The chat box is a light gray rectangle with a thin border. Inside, there are three messages, each on a new line and separated by a thin horizontal line. The first message is from 'kavin' and says 'hello'. The second message is also from 'kavin' and says 'hey, I'm testing out group messaging.'. The third message is from 'rahul' and says 'it's working'. Below the messages is a text input field with the placeholder text 'Message'. At the bottom right of the chat box is a green button with the text 'Send' in white.

kavin:hello

kavin:hey, I'm testing out group messaging.

rahul:it's working

Message

Send

4. We have also designed good UI in all the modules

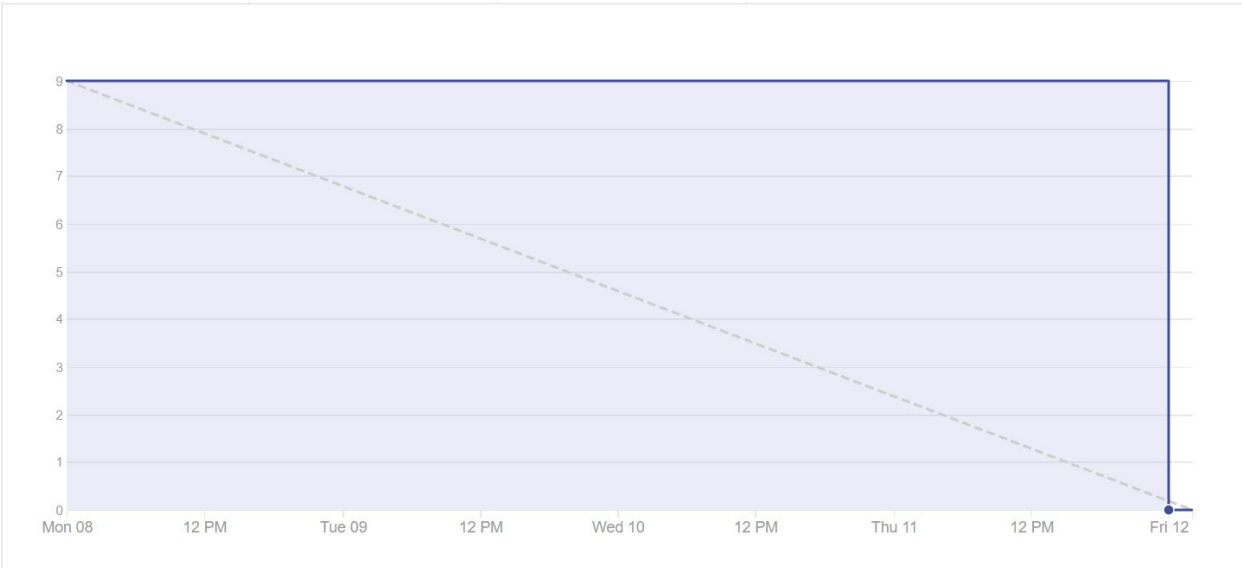
5. Heroku Deployment

We have also deployed our application in heroku link :

<https://ase5551-lab-3.herokuapp.com/>

Zenhub Tool

Burndown Charts:-

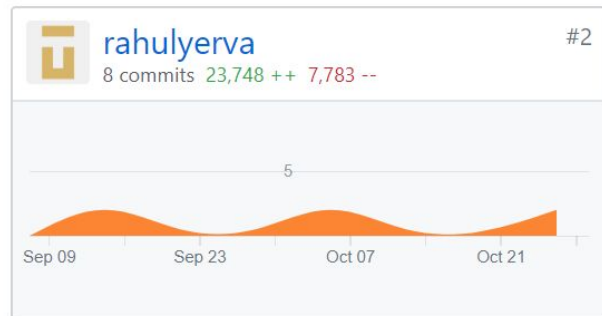
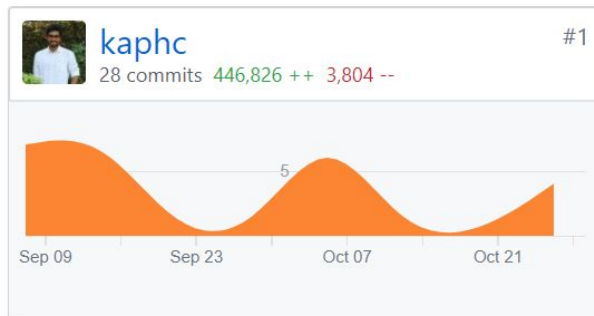
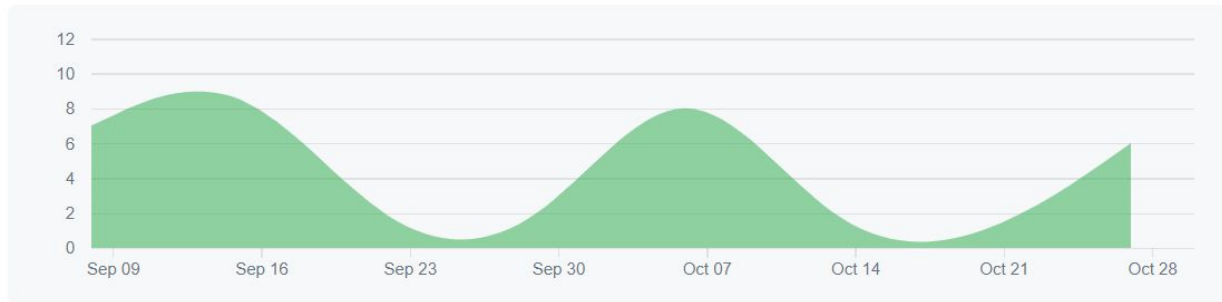


Contributors Chart:

Sep 9, 2018 – Oct 31, 2018

Contributions: Commits ▾

Contributions to master, excluding merge commits



Video

Link to code execution - [here](#)

Contribution By

Rahul Reddy Yerva

- Created mongodb database collection.
- Added REST service functionality.
- Heroku deployment

Kavin Kumar Arumugam

- Added socket.io node module
- Wiki page creation and documentation
- Integration of over all application