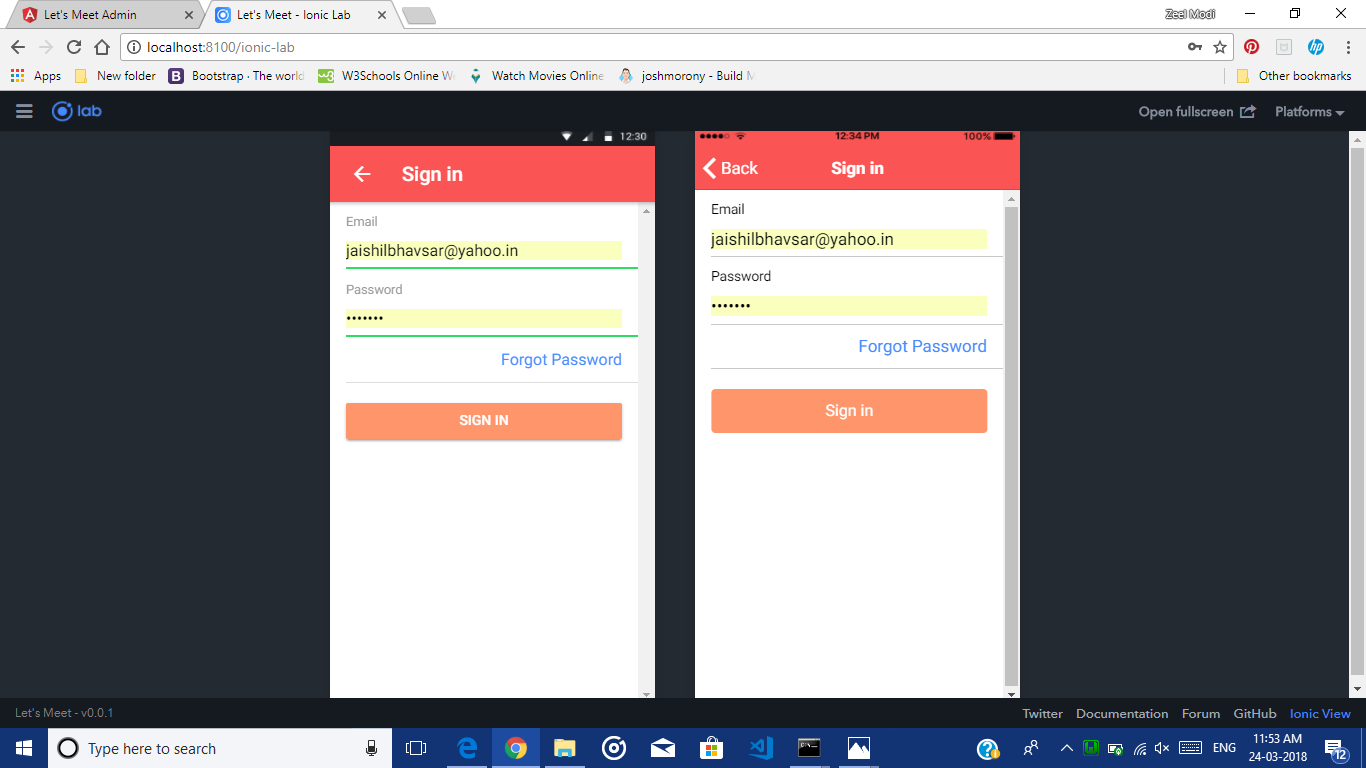
Let’s Meet

3005 – Jaishil Bhavsar

3036 – Zeel Modi

3186 – Bhargavi Jansari

# **Login**



**HTML**

<ion-header>

<ion-navbar color="signcolor">

<ion-title>{{ 'LOGIN\_TITLE' | translate }}</ion-title>

</ion-navbar>

</ion-header>

<ion-content>

<ion-list>

<ion-item>

<ion-label floating>{{ 'EMAIL' | translate }}</ion-label>

<ion-input type="email" [(ngModel)]="eid" name="email" ></ion-input>

</ion-item>

<!--

Want to use a Username instead of an Email? Here you go:

<ion-item>

<ion-label floating>{{ 'USERNAME' | translate }}</ion-label>

<ion-input type="text" [(ngModel)]="account.username" name="username"></ion-input>

</ion-item>

-->

<ion-item>

<ion-label floating>{{ 'PASSWORD' | translate }}</ion-label>

<ion-input type="password" [(ngModel)]="pass" name="password" ></ion-input>

</ion-item>

<ion-item >

<a (click)="showPrompt()" class="forgetAlign">Forgot Password</a>

</ion-item>

<div padding>

<button ion-button color="meetup" (click)="onClick()" block>{{ 'LOGIN\_BUTTON' | translate }}</button>

</div>

</ion-list>

</ion-content>

**Typescript**

import { Storage } from '@ionic/storage';

import { AlertController } from 'ionic-angular';

import { loginclass } from './loginclass';

import { user\_class } from './user\_class';

import { Component } from '@angular/core';

import { TranslateService } from '@ngx-translate/core';

import { IonicPage, NavController, ToastController } from 'ionic-angular';

import { User } from '../../providers/providers';

import { LoginproProvider } from '../../providers/loginpro/loginpro';

import { MainPage } from "../pages";

import { LoadingController } from 'ionic-angular/components/loading/loading-controller';

import { email\_class } from '../../shared/email\_class';

@IonicPage()

@Component({

selector: 'page-login',

templateUrl: 'login.html'

})

export class LoginPage {

// The account fields for the login form.

// If you're using the username field with or without email, make

// sure to add it to the type

/\*account: { email: string, password: string } = {

email: 'test@example.com',

password: 'test'

};\*/

// Our translated text strings

private loginErrorString: string;

u1: user\_class[] = [];

loginsession: loginclass;

constructor(public load: LoadingController, public alert: AlertController, public storage: Storage, public data: LoginproProvider, public navCtrl: NavController,

public user: User,

public toastCtrl: ToastController,

public translateService: TranslateService) {

this.translateService.get('LOGIN\_ERROR').subscribe((value) => {

this.loginErrorString = value;

})

}

// Attempt to login in through our User service

/\*dologin() {

this.data.doLogin(this.eid,this.pass).subscribe((resp) => {

this.navCtrl.push(MainPage);

}, (err) => {

this.navCtrl.push(MainPage);

// Unable to log in

let toast = this.toastCtrl.create({

message: this.loginErrorString,

duration: 3000,

position: 'top'

});

toast.present();

});

}\*/

dt: user\_class;

eid: string = "jaishilbhavsar@yahoo.in";

pass: string = "jaishil";

onClick() {

this.data.doLogin(this.eid, this.pass).subscribe(

(dt) => {

if (dt == "") {

let toast = this.toastCtrl.create({

message: "Invalid UserName Or Password , Try Again",

duration: 4000,

position: 'top'

});

toast.present();

}

else {

this.storage.set('uid', this.eid);

/\*this.storage.get('uid').then((val) => {

console.log(val);

});\*/

this.navCtrl.push(MainPage);

}

},

(err) => alert("invalid"),

);

}

forid: string = '';

u: user\_class[] = [];

validate() {

let l1 = this.load.create({

content: "Loading ..."

});

l1.present();

this.data.getUser(this.forid)

.subscribe(

(dt:any) => {

if(dt!="")

{

this.u = dt;

this.eid = this.u[0].user\_id;

var message = "Hello " + this.forid + ". You have requested to reset the password. your password is '" + this.u[0].user\_pass + "'. Password is one of the confidential thing, Don't share it with anyone.";

this.data.sendMail(new email\_class(message, this.forid, "Resetting the password of Lets\_Meet."))

.subscribe(

(data1: any) => {

console.log("mail sent");

alert("The Password has been sent to " + this.forid);

},

function (e) {

alert(e);

},

function () {

l1.dismiss();

}

);

}

else{

alert("enter valid mail");

}

}

,

function (e) {

alert(e);

},

function () {

l1.dismiss();

}

);

}

showPrompt() {

let prompt = this.alert.create({

title: 'Forgot Password',

message: "Enter Your Email Id To Get Your Password",

inputs: [

{

name: 'name',

placeholder: 'Email\_id'

},

],

buttons: [

{

text: 'Cancel',

handler: data => {

console.log('Cancel clicked');

}

},

{

text: 'Send',

handler: data => {

this.forid = data.name;

this.validate();

}

}

]

});

prompt.present();

}

}

**NodeJs**

var db = require('../dbconnection');

var u={

getUser:function(us,callback)

{

console.log(us.user\_id);

console.log(us.user\_pass);

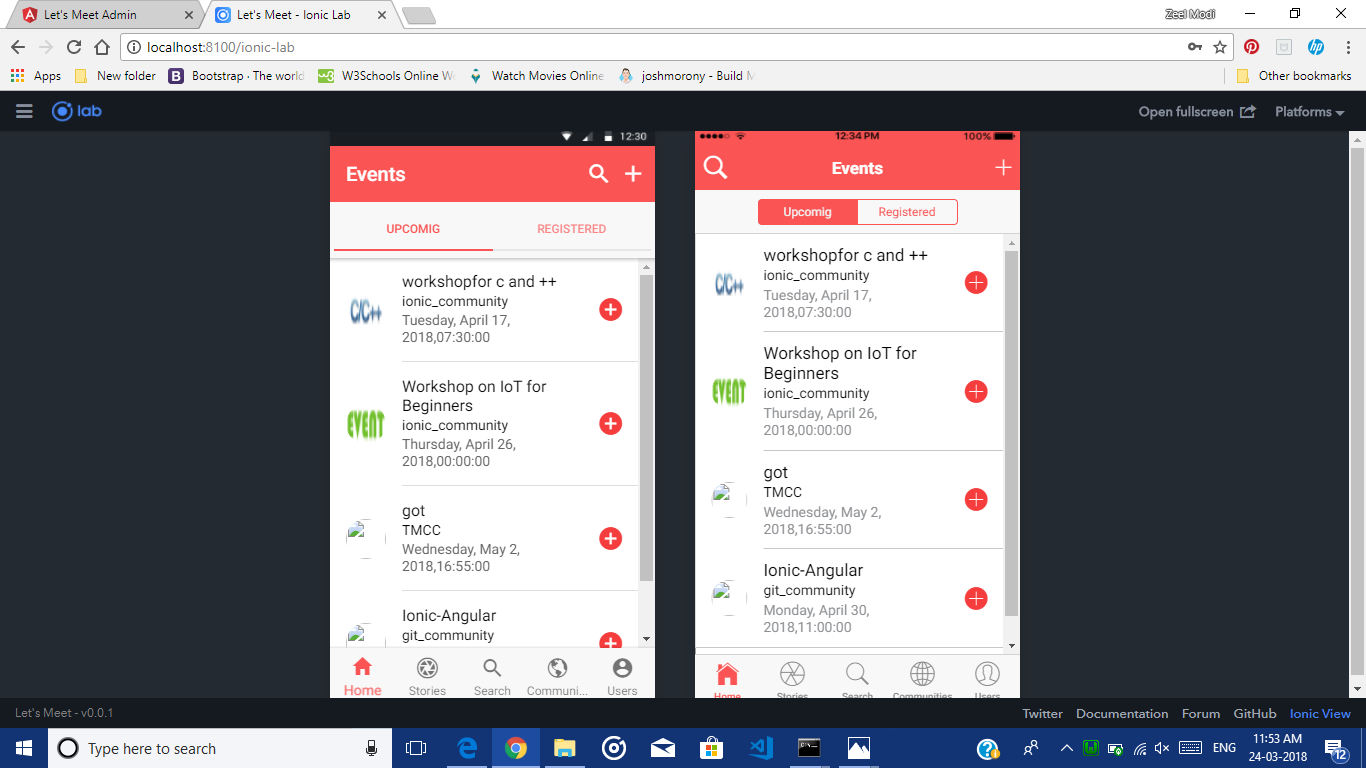
return db.query("select \* from user\_tbl where user\_id=? and user\_pass=?", [us.user\_id,us.user\_pass], callback);

}

};

module.exports=u;

# **Events**



**HTML**

<ion-header>

<ion-navbar color="signcolor">

<!--<ion-title>{{ 'LIST\_MASTER\_TITLE' | translate }}</ion-title>-->

<ion-title>Events</ion-title>

<ion-buttons start>

<button (click)="onSearchIcon()" ion-button icon-only>

<ion-icon name='search'></ion-icon>

</button>

</ion-buttons>

<ion-buttons end>

<button ion-button icon-only (click)="addItem()">

<ion-icon name="add"></ion-icon>

</button>

</ion-buttons>

</ion-navbar>

<ion-toolbar no-border-top>

<ion-segment [(ngModel)]="event">

<ion-segment-button value="upcEvents">

Upcomig

</ion-segment-button>

<ion-segment-button value="rgEvents">

Registered

</ion-segment-button>

</ion-segment>

</ion-toolbar>

</ion-header>

<ion-content>

<ion-searchbar \*ngIf="flag" (ionInput)="getItems($event)"></ion-searchbar>

<div [ngSwitch]="event">

<ion-list \*ngSwitchCase="'upcEvents'">

<ion-list>

<ion-item \*ngFor="let item of arr" class="text-wrap">

<ion-avatar item-start>

<img src="{{item.event\_pic}}" (click)="onClick(item.event\_id)">

</ion-avatar>

<h2 class="text-wrap" (click)="onClick(item.event\_id)">{{item.event\_name}}</h2>

<h3 class="text-wrap" (click)="onClick(item.event\_id)">{{item.comm\_name}}</h3>

<p class="text-wrap" (click)="onClick(item.event\_id)">{{item.event\_date | date:'fullDate'}},{{item.event\_s\_time}}</p>

<ion-icon name="add-circle" (click)="onClickRSVP(item.event\_id)" item-end color="danger"></ion-icon>

<!--<button ion-button (click)="onRemoveRSVP(item.event\_id)" \*ngIf="going\_button[i]" clear item-end>Going</button>-->

</ion-item>

</ion-list>

</ion-list>

<ion-list \*ngSwitchCase="'rgEvents'">

<ion-list>

<ion-item \*ngFor="let item of reg" class="text-wrap">

<ion-avatar item-start>

<img src="{{item.event\_pic}}" (click)="onClick(item.event\_id)">

</ion-avatar>

<h2 class="text-wrap" (click)="onClick(item.event\_id)">{{item.event\_name}}</h2>

<h3 class="text-wrap" (click)="onClick(item.event\_id)">{{item.comm\_name}}</h3>

<p class="text-wrap" (click)="onClick(item.event\_id)">{{item.event\_date | date:'fullDate'}},{{item.event\_s\_time}}</p>

<ion-icon name="checkmark-circle" (click)="onClickRSVP(item.event\_id)" item-end color="danger"></ion-icon>

</ion-item>

</ion-list>

</ion-list>

</div>

</ion-content>

**Typescript**

import { Component } from '@angular/core';

import { LoadingController, ToastController, IonicPage, ModalController, NavController } from 'ionic-angular';

import { Storage } from "@ionic/storage";

import { Item } from '../../models/item';

import { Items } from '../../providers/providers';

import { Events\_Class } from "../../shared/event\_class";

import { EventDbProvider } from "../../providers/event-db/event-db";

import { ViewEventPage } from "../view-event/view-event";

import { EventCommunityDbProvider } from "../../providers/event-community-db/event-community-db";

import { Event\_Comm\_Rsvp } from "../../shared/event\_community\_rsvp\_class";

import { RSVP\_Class } from '../../shared/rsvp\_class';

import { RsvpDbProvider } from '../../providers/rsvp-db/rsvp-db';

import { Event\_Community\_Class } from "../../shared/event\_community\_class";

@IonicPage()

@Component({

selector: 'page-list-master',

templateUrl: 'list-master.html'

})

export class ListMasterPage {

event: string = "upcEvents";

currentItems: Item[];

u\_id: string = "";

arr: Event\_Community\_Class[] = [];

arr1: Event\_Community\_Class[] = [];

arrRsvp: RSVP\_Class;

//arr1: Events[] = [];

//join\_button: boolean[] = [true];

//going\_button: boolean[] = [false];

user\_id: string = "";

flag: boolean = false;

txtsearch: string = '';

reg: Event\_Comm\_Rsvp[] = [];

constructor(public storage: Storage,

public tos: ToastController,

public \_data: EventDbProvider,

public \_data1: EventCommunityDbProvider,

public \_dataRSVP: RsvpDbProvider,

public load: LoadingController,

public navCtrl: NavController,

public items: Items,

public modalCtrl: ModalController) {

}

ionViewDidLoad() {

let l1 = this.load.create({

content: 'Loading Events'

});

l1.present();

this.storage.get('uid').then((val) => {

this.u\_id = val;

//this.\_data1.getAllEventsAndCommunities().subscribe(

this.\_data.getNotRegisteredEventsofUser(this.u\_id).subscribe(

(d: Event\_Community\_Class[]) => {

this.arr = d;

this.arr1 = d;

console.log(this.arr);

},

function (e) {

alert(e);

},

function () {

l1.dismiss();

}

);

});

let l2 = this.load.create({

content: 'Registered Events'

});

l2.present();

this.storage.get('uid').then((val) => {

this.u\_id = val;

this.\_data.getRegisteredEventsofUser(this.u\_id).subscribe(

(data: Event\_Comm\_Rsvp[]) => {

this.reg = data;

console.log(this.reg);

},

function (err) {

alert(err);

},

function () {

l2.dismiss();

}

);

});

}

ionViewDidEnter(){

this.ionViewDidLoad();

}

/\*\*

\* Prompt the user to add a new item. This shows our ItemCreatePage in a

\* modal and then adds the new item to our data source if the user created one.

\*/

addItem() {

let addModal = this.modalCtrl.create('ItemCreatePage');

addModal.onDidDismiss(item => {

this.ionViewDidLoad();

})

addModal.present();

//this.navCtrl.push(CreateEventPage);

}

onClick(event\_id: Events\_Class) {

this.navCtrl.push(ViewEventPage, { e\_id: event\_id });

//this.navCtrl.push(ViewEventPage);

this.storage.set('evn\_id', event\_id);

}

onRSVP(event\_id) {

this.storage.get('uid').then((val) => {

this.user\_id = val;

let l1 = this.load.create({

content: 'Joining ...'

});

l1.present();

let t1 = this.tos.create({

duration: 3000,

message: "Joined ..."

})

this.\_dataRSVP.addRSVP(new RSVP\_Class(null, this.user\_id, event\_id)).subscribe(

(data: any) => {

this.ionViewDidLoad();

//this.join\_button = false;

//this.going\_button = true;

},

function (e) {

alert(e);

},

function () {

l1.dismiss();

t1.present();

}

);

});

}

onSearchIcon() {

if (this.flag == true) {

this.flag = false;

}

else {

this.flag = true;

}

}

getItems(ev) {

this.arr = this.arr1;

var val = ev.target.value;

if (val && val.trim() != '') {

this.arr = this.arr.filter((x) =>

x.event\_name.toLocaleLowerCase().indexOf(val.toLocaleLowerCase()) > -1);

}

}

onClickRSVP(evn\_id) {

this.storage.get('uid').then((val) => {

this.user\_id = val;

let l1 = this.load.create({

content: 'Joining ...'

});

l1.present();

this.\_dataRSVP.checkRSVPOfEvent(this.u\_id, evn\_id).subscribe(

(data) => {

if (data == "") {

this.onRSVP(evn\_id);

//console.log("hello");

//console.log(data);

}

else {

let t1 = this.tos.create({

duration: 3000,

message: "You're already Going ...",

position: 'top'

});

t1.present();

}

},

function (e) {

alert(e);

},

function () {

l1.dismiss();

}

);

});

}

}

**NodeJs**

var db = require('../dbconnection');

var fs = require('fs');

var Event = {

getAllEvent: function (callback) {

return db.query("SELECT e.\*,u.\* from event\_tbl e,user\_tbl u WHERE e.fk\_user\_id=u.user\_id", callback);

},

getUpcomingEvents: function (callback) {

return db.query("select \* from event\_tbl where event\_date > CURDATE()", callback);

},

getEventById: function (id, callback) {

return db.query("select \* from event\_tbl where event\_id=?", [id], callback);

},

addEvent: function (e, filename, callback) {

return db.query("insert into event\_tbl values(?,?,?,?,?,?,?,?,?,?,?)", [null, e.event\_name, e.event\_des, filename, e.event\_s\_time, e.event\_e\_time, e.event\_date, e.event\_loc, e.fk\_user\_id, e.fk\_comm\_id, "true"], callback);

//return db.query("insert into event\_tbl (event\_name,event\_des,event\_pic,event\_s\_time,event\_e\_time,event\_date,event\_loc,fk\_user\_id,fk\_comm\_id,verify) values(?,?,?,?,?,?,?,?,?,?)",[e.event\_name,e.event\_des,e.event\_pic,e.event\_s\_time,e.event\_e\_time,e.event\_date,e.event\_loc,e.fk\_user\_id,e.fk\_comm\_id],callback);

},

deleteEvent: function (id, callback) {

var evn = db.query("select \* from event\_tbl where event\_id=?", [id]);

evn.on('result', function (row) {

console.log(row.event\_pic);

if (row.event\_pic != '') {

var path = 'public/images/events/' + row.event\_pic;

console.log(path);

fs.unlink(path, function (err) {

if (err) {

console.log(err);

}

console.log('Deleted successfuly');

});

}

});

return db.query("delete from event\_tbl where event\_id=?", [id], callback);

},

updateEvent: function (id, e, filename, callback) {

var evn = db.query("select \* from event\_tbl where event\_id=?", [id]);

evn.on('result', function (row) {

console.log(row.event\_pic);

if (row.post\_pic != '') {

var path = 'public/images/posts/' + row.event\_pic;

console.log(path);

fs.unlink(path, function (err) {

if (err) {

console.log(err);

}

console.log('Deleted successfuly');

});

}

});

return db.query("update event\_tbl set event\_name=?,event\_des=?,event\_pic=?,event\_s\_time=?,event\_e\_time=?,event\_date=?,event\_loc=?,fk\_comm\_id=? where event\_id=?", [e.event\_name, e.event\_des, filename, e.event\_s\_time, e.event\_e\_time, e.event\_date, e.event\_loc, e.fk\_comm\_id, id], callback);

},

eventRegisteredRsvp: function (user\_id, callback) {

//SELECT DISTINCT e.\*,r.\*,c.\* from event\_tbl e,rsvp\_tbl r,communities\_tbl c WHERE c.comm\_id=e.fk\_comm\_id AND e.event\_id=r.fk\_event\_id and NOT r.rsvp\_fk\_user\_id='zeel91297@gmail.com'

return db.query("select e.\*,r.\*,c.\* from event\_tbl e,rsvp\_tbl r,communities\_tbl c where r.fk\_event\_id=e.event\_id and c.comm\_id=e.fk\_comm\_id and r.rsvp\_fk\_user\_id=? and e.event\_date > CURDATE() ", [user\_id], callback);

},

getAllUpcNotReg: function (user\_id, callback) {

return db.query("SELECT e.\*,r.\*,c.\* from event\_tbl e,rsvp\_tbl r,communities\_tbl c where e.event\_id=r.fk\_event\_id AND e.fk\_comm\_id=c.comm\_id AND r.rsvp\_fk\_user\_id!=? and e.event\_date > CURDATE()", [user\_id], callback);

}

};

module.exports = Event;

# **Create Event**

# 

**HTML**

<ion-header>

<ion-navbar>

<ion-title>Create Event</ion-title>

<ion-buttons start>

<button ion-button (click)="cancel()">

<span color="primary" showWhen="ios">

{{ 'CANCEL\_BUTTON' | translate }}

</span>

<ion-icon name="md-close" showWhen="android,windows"></ion-icon>

</button>

</ion-buttons>

<ion-buttons end>

<button ion-button (click)="done()" [disabled]="!isReadyToSave" strong>

<span color="primary" showWhen="ios">

{{ 'DONE\_BUTTON' | translate }}

</span>

<ion-icon name="md-checkmark" showWhen="core,android,windows"></ion-icon>

</button>

</ion-buttons>

</ion-navbar>

</ion-header>

<ion-content padding>

<form \*ngIf="form" [formGroup]="form" (ngSubmit)="addItem()">

<input type="file" #fileInput style="visibility: hidden; height: 0px" name="files[]" (change)="processWebImage($event)" />

<div class="profile-image-wrapper" (click)="getPicture()">

<div class="profile-image-placeholder" \*ngIf="!this.form.controls.profilePic.value">

<ion-icon name="add"></ion-icon>

<div>

{{ 'ITEM\_CREATE\_CHOOSE\_IMAGE' | translate }}

</div>

</div>

<div class="profile-image" [style.backgroundImage]="getProfileImageStyle()" \*ngIf="this.form.controls.profilePic.value"></div>

</div>

<ion-list>

<ion-item>

<ion-input type="text" formControlName="event\_name" [(ngModel)]="event\_name" placeholder="Event Name"></ion-input>

</ion-item>

<ion-item>

<ion-textarea placeholder="Event Description" formControlName="event\_des" [(ngModel)]="event\_des"></ion-textarea>

</ion-item>

<ion-item>

<ion-label>Event Starting Time</ion-label>

<ion-datetime displayFormat="hh:mm A" formControlName="event\_s\_time" pickerFormat="hh mm A" [(ngModel)]="event\_s\_time"></ion-datetime>

</ion-item>

<ion-item>

<ion-label>Event Ending Time</ion-label>

<ion-datetime displayFormat="hh:mm A" formControlName="event\_e\_time" pickerFormat="hh mm A" [(ngModel)]="event\_e\_time"></ion-datetime>

</ion-item>

<ion-item>

<ion-label>Event Date</ion-label>

<ion-datetime displayFormat="DD/MMMM/YYYY" formControlName="event\_date" pickerFormat="DD MMMM YYYY" [(ngModel)]="event\_date"></ion-datetime>

</ion-item>

<ion-item>

<ion-label>Event Location</ion-label>

<ion-textarea formControlName="event\_loc" [(ngModel)]="event\_loc"></ion-textarea>

</ion-item>

<ion-item>

<ion-label>Community Name : </ion-label>

<ion-select formControlName="community\_id" [(ngModel)]="community\_id">

<ion-option \*ngFor="let item of arr" value="{{item.comm\_id}}">

{{item.comm\_name}}

</ion-option>

</ion-select>

</ion-item>

<!--<button ion-button block (click)="onAdd()" icon-right color="danger">

Create

<ion-icon name="checkmark-circle" color="primary"></ion-icon>

</button>-->

</ion-list>

</form>

</ion-content>

**Typescript**

import { Component, ViewChild } from '@angular/core';

import { FormBuilder, FormGroup, Validators } from '@angular/forms';

import { Camera } from '@ionic-native/camera';

import { IonicPage, NavController, ViewController, LoadingController, ToastController } from 'ionic-angular';

import { DateTime } from 'ionic-angular/components/datetime/datetime';

import { Storage } from "@ionic/storage";

import { HttpClient, HttpHeaders } from "@angular/common/http";

import { EventDbProvider } from "../../providers/event-db/event-db";

import { Events\_Class } from "../../shared/event\_class";

import { Community\_Class } from "../settings/community\_class";

import { ComminityDbTsProvider } from "../../providers/community-db/community-db";

//import { DateValidator } from "./event\_date";

@IonicPage()

@Component({

selector: 'page-item-create',

templateUrl: 'item-create.html'

})

export class ItemCreatePage {

@ViewChild('fileInput') fileInput;

isReadyToSave: boolean;

item: any;

form: FormGroup;

arr: Community\_Class[] = [];

event\_id: any = null;

event\_name: string = "";

event\_des: string = "";

event\_s\_time: any;

event\_e\_time: any;

event\_date: any;

event\_loc: string = "";

created\_by: string = "";

event\_pic: string = "";

community\_id: any;

event\_verify: any = "true";

selectedFile: File = null;

constructor(public http: HttpClient

, public storage: Storage,

public \_data: EventDbProvider,

public \_data1: ComminityDbTsProvider,

public load: LoadingController,

public tos: ToastController,

public navCtrl: NavController,

public viewCtrl: ViewController,

formBuilder: FormBuilder,

public camera: Camera) {

this.form = formBuilder.group({

profilePic: [''],

event\_name: ['', Validators.required],

event\_des: ['', Validators.compose([Validators.minLength(15), Validators.required])],

event\_s\_time: ['', Validators.required],

event\_e\_time: ['', Validators.required],

event\_date: ['', Validators.required],

event\_loc: ['', Validators.compose([Validators.minLength(10), Validators.required])],

community\_id: ['', Validators.required]

});

// Watch the form for changes, and

this.form.valueChanges.subscribe((v) => {

this.isReadyToSave = this.form.valid;

});

}

ionViewDidLoad() {

this.\_data1.getAllCommunities().subscribe(

(data: Community\_Class[]) => {

this.arr = data;

},

function (e) {

alert(e);

},

function () {

}

)

}

getPicture() {

if (Camera['installed']()) {

this.camera.getPicture({

destinationType: this.camera.DestinationType.DATA\_URL,

targetWidth: 96,

targetHeight: 96

}).then((data) => {

this.form.patchValue({ 'profilePic': 'data:image/jpg;base64,' + data });

}, (err) => {

alert('Unable to take photo');

})

} else {

this.fileInput.nativeElement.click();

}

}

processWebImage(event) {

let reader = new FileReader();

reader.onload = (readerEvent) => {

let imageData = (readerEvent.target as any).result;

this.form.patchValue({ 'profilePic': imageData });

};

reader.readAsDataURL(event.target.files[0]);

this.selectedFile = <File>event.target.files[0];

//alert(this.selectedFile.type);

if (this.selectedFile.type != 'image/png' && this.selectedFile.type != 'image/jpeg') {

this.selectedFile = null;

this.isReadyToSave = this.form.invalid;

const toast = this.tos.create({

message: 'Only Image formats are accepted!',

showCloseButton: true,

closeButtonText: 'Ok'

});

toast.present();

}

}

getProfileImageStyle() {

return 'url(' + this.form.controls['profilePic'].value + ')'

}

/\*\*

\* The user cancelled, so we dismiss without sending data back.

\*/

cancel() {

this.viewCtrl.dismiss();

}

/\*\*

\* The user is done and wants to create the item, so return it

\* back to the presenter.

\*/

done() {

if (!this.form.valid) { return; }

this.storage.get('uid').then((val) => {

this.created\_by = val;

let l1 = this.load.create({

content: 'Creating ...'

});

l1.present();

let t1 = this.tos.create({

duration: 3000,

message: "Added ..."

})

const fd = new FormData();

fd.append("event\_id", this.event\_id);

fd.append("event\_name", this.event\_name);

fd.append("event\_des", this.event\_des);

fd.append("image", this.selectedFile, this.selectedFile.name);

fd.append("event\_s\_time", this.event\_s\_time);

fd.append("event\_e\_time", this.event\_e\_time);

fd.append("event\_date", this.event\_date);

fd.append("event\_loc", this.event\_loc);

fd.append("fk\_user\_id", this.created\_by);

fd.append("fk\_comm\_id", this.community\_id);

fd.append("event\_verify", "true");

this.\_data.addEvent(fd).subscribe(

(data: any) => {

console.log(data);

this.viewCtrl.dismiss();

t1.present();

},

function (e) {

alert(e);

},

function () {

l1.dismiss();

}

)

});

// this.viewCtrl.dismiss(this.form.value);

}

}

**SCSS**

page-item-create {

.profile-image-wrapper {

text-align: center;

margin: 20px 0;

.profile-image {

width: 96px;

height: 96px;

border-radius: 50%;

display: inline-block;

background-repeat: no-repeat;

background-size: cover;

background-position: center;

}

.profile-image-placeholder {

display: inline-block;

background-color: #eee;

width: 96px;

height: 96px;

border-radius: 50%;

font-size: 12px;

ion-icon {

font-size: 44px;

margin-bottom: -10px;

margin-top: 10px;

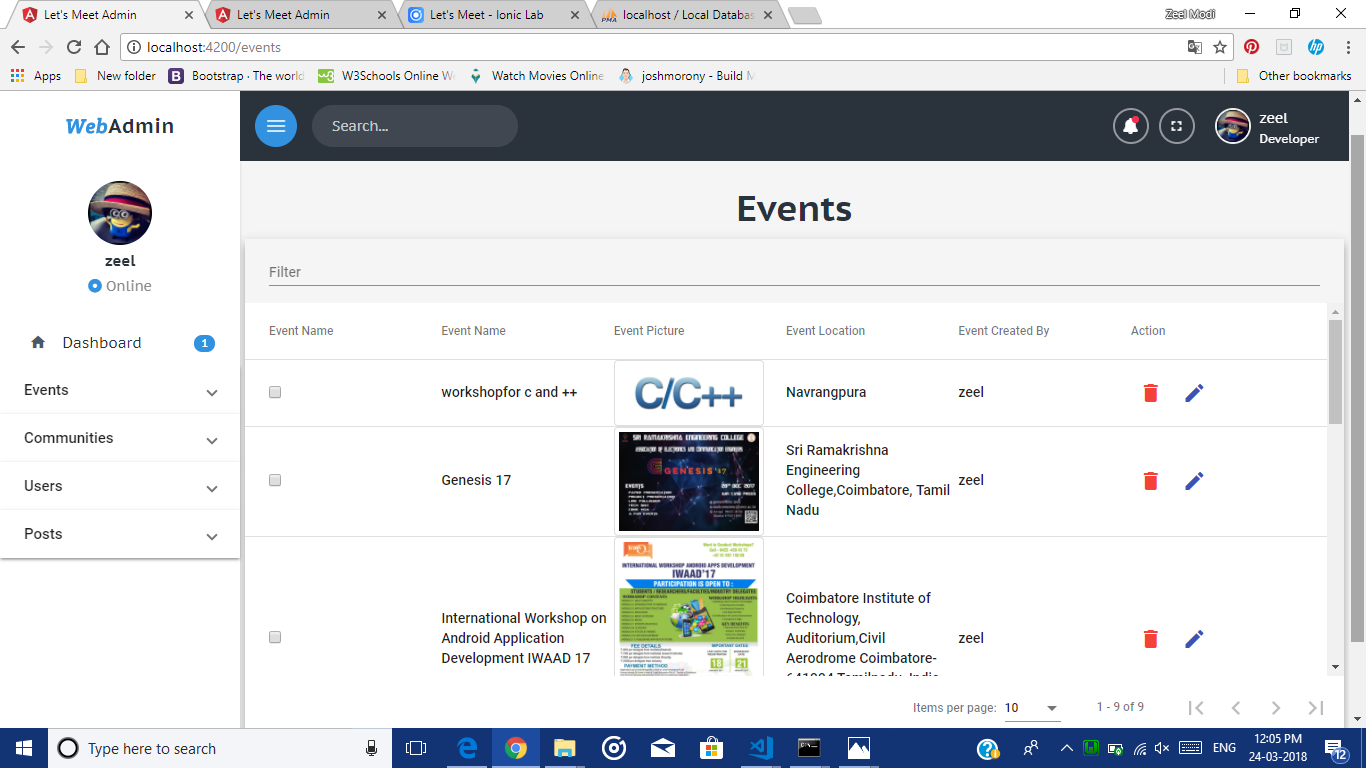
}

}

}

}

# **View Events [Admin]**



**HTML**

<p>

events works!

</p>

<app-header></app-header>

<div class="content-page">

<div class="content page-content-wrapper">

<h1 class="display-1" style="text-align:center">Events</h1>

<!--<div class="">

<div class="page-header-title">

<h4 class="page-title">Events</h4>

</div>-->

<!--<div class="page-content-wrapper ">-->

<div class="example-container mat-elevation-z8">

<div class="example-header">

<mat-form-field>

<input matInput (keyup)="applyFilter($event.target.value)" placeholder="Filter">

</mat-form-field>

</div>

<mat-table #table [dataSource]="dataSource" matSort>

<ng-container matColumnDef="check">

<mat-header-cell \*matHeaderCellDef mat-sort-header>Event Name </mat-header-cell>

<mat-cell \*matCellDef="let element">

<input type="checkbox" (change)="ceckChange(element)" name="usercheck">

</mat-cell>

</ng-container>

<!--- Note that these columns can be defined in any order.

The actual rendered columns are set as a property on the row definition" -->

<ng-container matColumnDef="event\_name">

<mat-header-cell \*matHeaderCellDef mat-sort-header>Event Name </mat-header-cell>

<mat-cell \*matCellDef="let element"> {{element.event\_name}} </mat-cell>

</ng-container>

<ng-container matColumnDef="event\_pic">

<mat-header-cell \*matHeaderCellDef mat-sort-header> Event Picture </mat-header-cell>

<mat-cell \*matCellDef="let element">

<img src="http://localhost:3000/images/events/{{element.event\_pic}}" height="150" width="150" alt="..." class="img-thumbnail"> </mat-cell>

</ng-container>

<ng-container matColumnDef="event\_loc">

<mat-header-cell \*matHeaderCellDef mat-sort-header> Event Location </mat-header-cell>

<mat-cell \*matCellDef="let element"> {{element.event\_loc}} </mat-cell>

</ng-container>

<ng-container matColumnDef="fk\_user\_id">

<mat-header-cell \*matHeaderCellDef mat-sort-header> Event Created By </mat-header-cell>

<mat-cell \*matCellDef="let element"> {{element.user\_name}} </mat-cell>

</ng-container>

<ng-container matColumnDef="action">

<mat-header-cell \*matHeaderCellDef mat-sort-header> Action </mat-header-cell>

<mat-cell \*matCellDef="let element">

<button (click)="deleteEvent(element)" color="warn" mat-icon-button>

<mat-icon>delete</mat-icon>

</button>

<button (click)="updateEvent(element)" color="primary" mat-icon-button>

<mat-icon>create</mat-icon>

</button>

</mat-cell>

</ng-container>

<mat-header-row \*matHeaderRowDef="displayedColumns"></mat-header-row>

<mat-row \*matRowDef="let row; columns: displayedColumns;" (click)="selection.toggle(row)"></mat-row>

</mat-table>

<mat-paginator #paginator [pageSize]="10" [pageSizeOptions]="[5, 10, 20]" [showFirstLastButtons]="true">

</mat-paginator>

</div>

<!--</div>-->

</div>

<footer class="footer"> © 2016 WebAdmin - All Rights Reserved. </footer>

**Typescript**

import { Component, OnInit, ViewChild, AfterViewInit } from '@angular/core';

import { Router } from '@angular/router';

import { DataSource } from '@angular/cdk/collections';

import { BehaviorSubject } from 'rxjs/BehaviorSubject';

import { Observable } from 'rxjs/Observable';

import { MatTableDataSource, MatSort, MatPaginator, Sort } from '@angular/material';

import { SelectionModel } from '@angular/cdk/collections';

import { Event\_User\_class } from '../shared/event\_user\_class';

import { EventsDbService } from '../providers/eventsDb/events-db.service';

@Component({

selector: 'app-events',

templateUrl: './events.component.html',

styleUrls: ['./events.component.css']

})

export class EventsComponent implements OnInit {

@ViewChild(MatPaginator) paginator: MatPaginator;

@ViewChild(MatSort) sort: MatSort;

displayedColumns = ['check', 'event\_name', 'event\_pic', 'event\_loc', 'fk\_user\_id', 'action'];

dataSource: MatTableDataSource<Event\_User\_class>;

selection = new SelectionModel<Event\_User\_class>(true, []);

arrEvents: Event\_User\_class[] = [];

constructor(public \_dataEvent: EventsDbService,

public router: Router) { }

ngOnInit() {

this.\_dataEvent.getAllEvents().subscribe(

(data: Event\_User\_class[]) => {

this.arrEvents = data;

console.log(data);

this.dataSource = new MatTableDataSource(this.arrEvents);

this.dataSource.paginator = this.paginator;

this.dataSource.sort = this.sort;

},

function (err) {

alert(err);

},

function () {

}

);

}

applyFilter(filterValue: string) {

filterValue = filterValue.trim(); // Remove whitespace

filterValue = filterValue.toLowerCase(); // MatTableDataSource defaults to lowercase matches

this.dataSource.filter = filterValue;

}

deleteEvent(item) {

this.\_dataEvent.deleteEvent(item).subscribe(

(data: any) => {

this.ngOnInit();

},

function (err) {

alert(err);

},

function () {

}

);

}

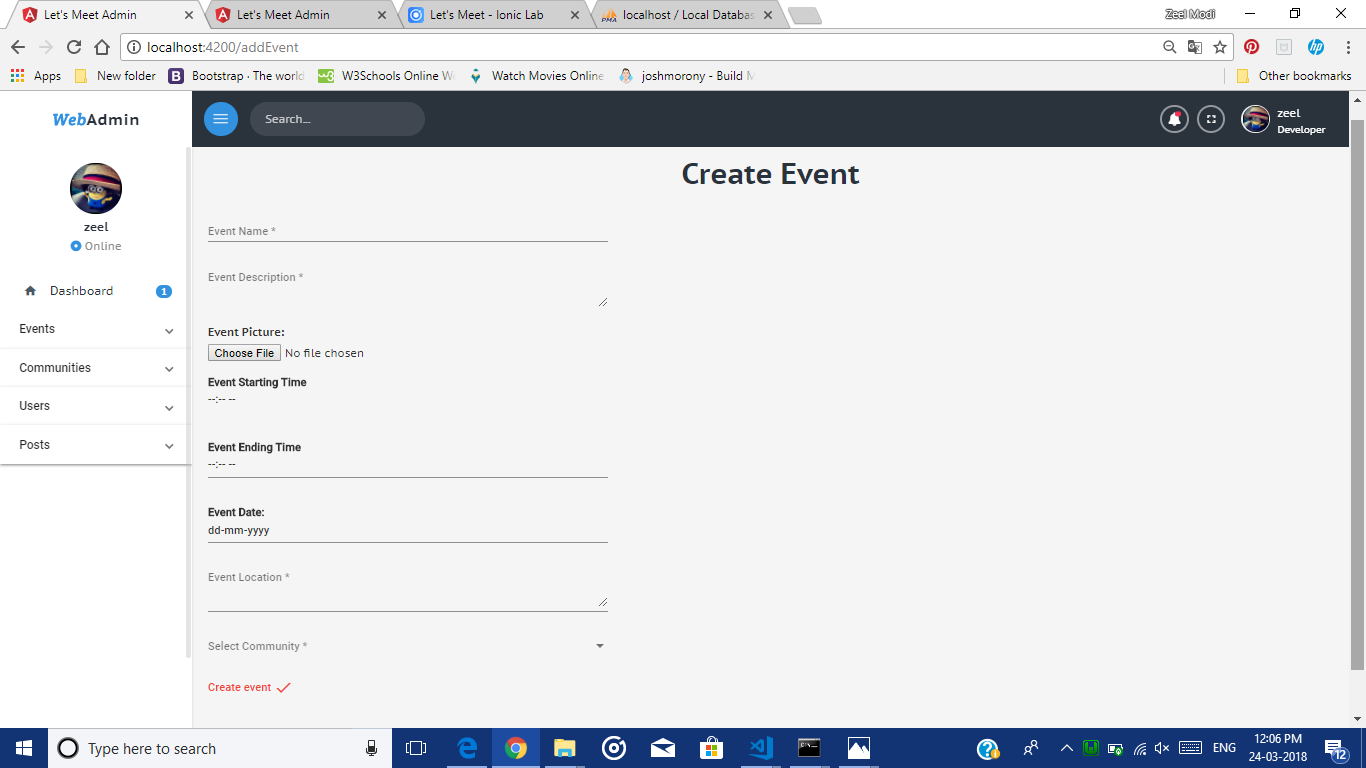
updateEvent(item) {

this.router.navigate(['/updateEvent', item.event\_id]);

}

}

# **Create Event [Admin]**



**HTML**

<!--<p>

add-event works!

</p>-->

<app-header></app-header>

<div class="content-page">

<div class="content page-content-wrapper">

<h1 class="display-1" style="text-align:center">Create Event</h1>

<div align="left" style="padding:15px">

<form class="example-form" (ngSubmit)="onAdd(eventform)" #eventform="ngForm">

<mat-form-field class="example-full-width">

<input matInput ngModel required placeholder="Event Name" name="event\_name" #event\_name="ngModel">

<mat-error \*ngIf="event\_name.hasError('required')">Please Enter Event Name</mat-error>

</mat-form-field>

<mat-form-field class="example-full-width">

<textarea matInput ngModel required placeholder="Event Description" name="event\_des" #event\_des="ngModel" cols="30" rows="3"></textarea>

<mat-error \*ngIf="event\_des.hasError('required')">Please Enter Event Description</mat-error>

</mat-form-field>

<!--<mat-form-field class="example-full-width">-->

<label>Event Picture:</label>

<input ngModel required type="file" name="event\_pic" #event\_pic="ngModel" (change)="onFileSelected($event)" id="">

<!--<mat-error \*ngIf="event\_pic.hasError('required')">Please Enter Event Picture</mat-error>-->

<mat-form-field class="example-full-width">

<label for="">Event Starting Time</label>

<input matInput ngModel required type="time" name="event\_s\_time" #event\_s\_time="ngModel">

<mat-error \*ngIf="event\_s\_time.hasError('required')">Please Enter Starting Time</mat-error>

</mat-form-field>

<mat-form-field class="example-full-width">

<label for="">Event Ending Time</label>

<input matInput ngModel type="time" name="event\_e\_time" #event\_e\_time="ngModel">

</mat-form-field>

<mat-form-field class="example-full-width">

<label for="">Event Date:</label>

<input matInput ngModel required type="date" name="event\_date" #event\_date="ngModel">

<mat-error \*ngIf="event\_date.hasError('required')">Please Enter Event Date</mat-error>

</mat-form-field>

<mat-form-field class="example-full-width">

<textarea matInput ngModel required placeholder="Event Location" name="event\_loc" #event\_loc="ngModel" cols="30" rows="3"></textarea>

<mat-error \*ngIf="event\_loc.hasError('required')">Please Enter Event Location</mat-error>

</mat-form-field>

<mat-form-field>

<mat-select required placeholder="Select Community" name="community\_id" [(ngModel)]="community\_id" #c\_id="ngModel">

<mat-option \*ngFor="let item of arrCommu" [value]="item.comm\_id">

{{ item.comm\_name }}

</mat-option>

</mat-select>

<mat-error \*ngIf="c\_id.hasError('required')">Please Select a Community</mat-error>

</mat-form-field>

<button type="submit" mat-button mat-icon-button color="warn">Create event

<mat-icon>done</mat-icon>

</button>

</form>

</div>

</div>

<!--</div>-->

</div>

<footer class="footer"> © 2016 WebAdmin - All Rights Reserved. </footer>

**Typescript**

import { Component, OnInit, ViewChild } from '@angular/core';

import { Router } from '@angular/router';

import { EventsDbService } from '../providers/eventsDb/events-db.service';

import { CommunityDbService } from '../providers/communitiesDb/community-db.service';

import { FormControl, Validators } from '@angular/forms';

import { routing } from '../app.routing';

import { Community\_Class } from '../shared/community\_class';

@Component({

selector: 'app-add-event',

templateUrl: './add-event.component.html',

styleUrls: ['./add-event.component.css']

})

export class AddEventComponent implements OnInit {

event\_id: any = null;

event\_name: any = '';

event\_des: any = '';

event\_s\_time: any;

event\_e\_time: any;

event\_date: any;

event\_loc: any = '';

created\_by: any = 'zeel91297@gmail.com';

event\_pic: any = '';

community\_id: any;

event\_verify: any = '';

selectedFile: File = null;

arrCommu: Community\_Class[] = [];

constructor(public \_dataEvent: EventsDbService,

public \_dataCommu: CommunityDbService,

public router: Router) { }

ngOnInit() {

this.\_dataCommu.getAllCommunities().subscribe(

(data: Community\_Class[]) => {

this.arrCommu = data;

},

function (e) {

alert(e);

},

function () {

}

);

}

onFileSelected(value) {

this.selectedFile = <File>value.target.files[0];

console.log(value);

}

onAdd(eventform) {

this.event\_name = eventform.value.event\_name;

this.event\_des = eventform.value.event\_des;

this.event\_s\_time = eventform.value.event\_s\_time;

this.event\_e\_time = eventform.value.event\_e\_time;

this.event\_date = eventform.value.event\_date;

this.event\_loc = eventform.value.event\_loc;

this.community\_id = eventform.value.community\_id;

const fd = new FormData();

fd.append('event\_id', this.event\_id);

fd.append('event\_name', this.event\_name);

fd.append('event\_des', this.event\_des);

fd.append('image', this.selectedFile, this.selectedFile.name);

fd.append('event\_s\_time', this.event\_s\_time);

fd.append('event\_e\_time', this.event\_e\_time);

fd.append('event\_date', this.event\_date);

fd.append('event\_loc', this.event\_loc);

fd.append('fk\_user\_id', this.created\_by);

fd.append('fk\_comm\_id', this.community\_id);

fd.append('event\_verify', 'true');

this.\_dataEvent.addEvent(fd).subscribe(

(data: any) => {

console.log(data);

this.router.navigate(['/events']);

},

function (e) {

alert(e);

},

function () {

}

);

}

}

**SCSS**

.example-form {

min-width: 150px;

max-width: 500px;

width: 100%;

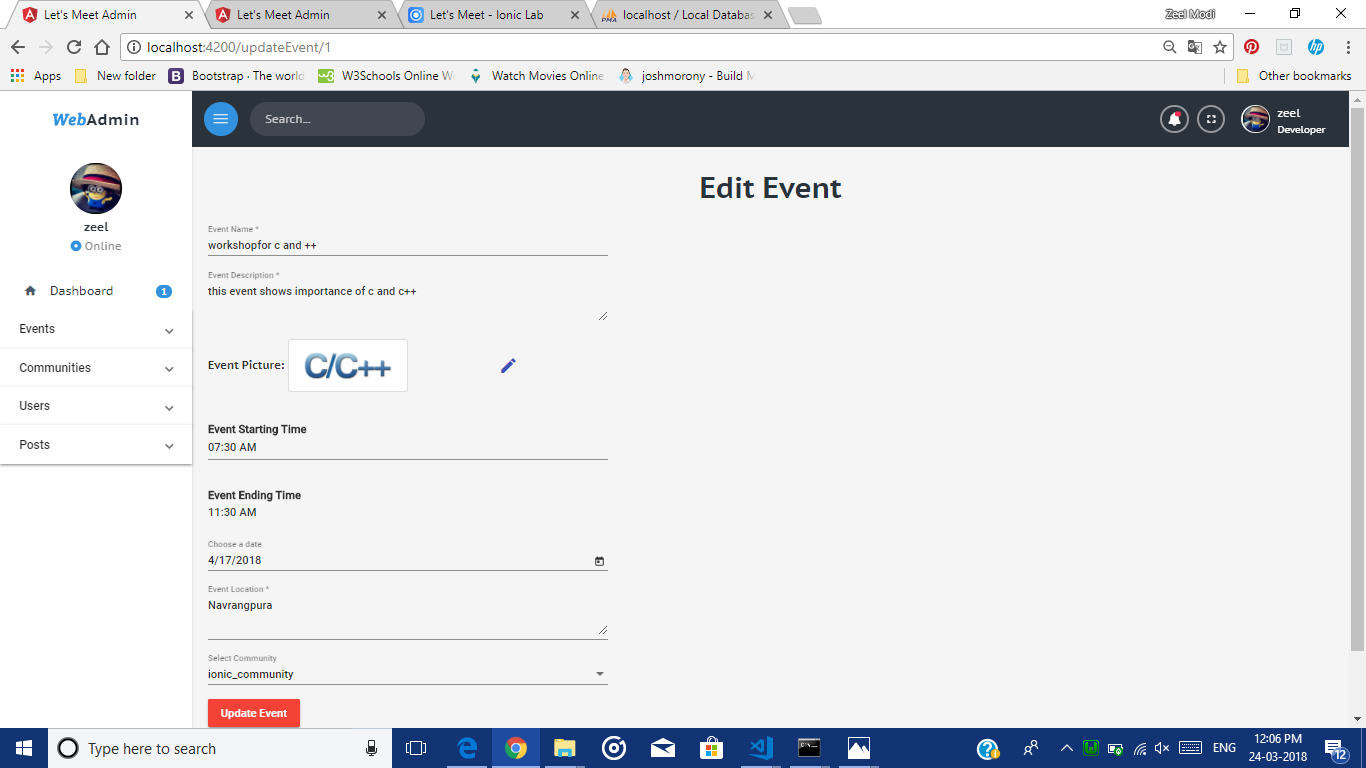
}

.example-full-width {

width: 100%;

}

# **Edit Event [Admin]**



**HTML**

<!-- <p>

update-post works!

</p> -->

<app-header></app-header>

<div class="content-page">

<div class="content page-content-wrapper">

<h1 class="display-1" style="text-align:center">Edit Post</h1>

<div align="left" style="padding:15px">

<form class="example-form" name="form11" (ngSubmit)="onUpdatePost(postForm)" #postForm="ngForm">

<mat-form-field class="example-full-width">

<input matInput required placeholder="Post Title" name="post\_title" #p\_title="ngModel" [(ngModel)]="post\_title">

<mat-error \*ngIf="p\_title.hasError('required')">Please Enter Post Title</mat-error>

</mat-form-field>

<mat-form-field class="example-full-width">

<textarea matInput required placeholder="Post Description" name="post\_des" #p\_des="ngModel" [(ngModel)]="post\_des" cols="30"

rows="3"></textarea>

<mat-error \*ngIf="p\_des.hasError('required')">Please Enter Post Description</mat-error>

</mat-form-field>

<label>Event Picture:</label>

<img src="http://localhost:3000/images/posts/{{post\_pic}}" height="150" width="150" alt="Not Available!" class="img-thumbnail">

<button mat-icon-button color="primary" name="btnImg">

<mat-icon (click)="getPicture()">mode edit</mat-icon>

</button>

<input required type="file" name="pst\_pic" style="visibility: hidden" #fileInput (change)="onFileSelected($event)"

id="">

<button type="submit" mat-button mat-raised-button color="warn">Update Post

<mat-icon>done</mat-icon>

</button>

</form>

</div>

</div>

</div>

<footer class="footer"> © 2016 WebAdmin - All Rights Reserved. </footer>

**Typescript**

import { Component, OnInit, ViewChild, ElementRef } from '@angular/core';

import { Router, ActivatedRoute } from '@angular/router';

import { FormControl, Validators } from '@angular/forms';

import { Subscription } from 'Rxjs/Rx';

import { routing } from '../app.routing';

import { PostsDbService } from '../providers/postsDb/posts-db.service';

import { Post\_Class } from '../shared/post\_class';

@Component({

selector: 'app-update-post',

templateUrl: './update-post.component.html',

styleUrls: ['./update-post.component.css']

})

export class UpdatePostComponent implements OnInit {

@ViewChild('fileInput') fileInput: ElementRef;

public \_subscription: Subscription;

post\_id: any;

arrPost: Post\_Class[] = [];

post\_title: string;

post\_des: string;

post\_pic: string;

selectedFile: File = null;

fileFlag: Boolean = false;

constructor(

public router: Router,

public \_dataPost: PostsDbService,

public activatedRoute: ActivatedRoute

) { }

ngOnInit() {

this.\_subscription = this.activatedRoute.params.subscribe(

(data: any) => {

this.post\_id = data['id'];

}

);

this.\_dataPost.getPostById(this.post\_id).subscribe(

(data: Post\_Class[]) => {

this.arrPost = data;

this.post\_title = this.arrPost[0].post\_title;

this.post\_des = this.arrPost[0].post\_des;

this.post\_pic = this.arrPost[0].post\_pic;

console.log(this.post\_pic);

},

function (err) {

alert(err);

},

function () {

}

);

}

onFileSelected(value) {

this.selectedFile = <File>value.target.files[0];

this.fileFlag = true;

console.log(value);

}

getPicture() {

this.fileInput.nativeElement.click();

}

onUpdatePost(postForm) {

this.post\_title = postForm.value.post\_title;

this.post\_des = postForm.value.post\_des;

if (this.selectedFile === null) {

} else {

const fd = new FormData();

fd.append('post\_id', this.post\_id);

fd.append('post\_title', this.post\_title);

fd.append('post\_des', this.post\_des);

fd.append('image', this.selectedFile, this.selectedFile.name);

this.\_dataPost.editPost(fd).subscribe(

(data: any) => {

console.log(data);

this.router.navigate(['/posts']);

},

function (err) {

alert(err);

},

function () {

}

);

}

}

}