CHAPTER 11

CODE

app.component.ts

```
import { Component, ViewChild, NgZone } from '@angular/core';
import { Platform, NavController, AlertController } from 'ionic-angular';
import { StatusBar } from '@ionic-native/status-bar';
import { SplashScreen } from '@ionic-native/splash-screen';
import { Login } from '../pages/login/login';
import { Newsfeed } from '../pages/newsfeed/newsfeed';
import { Chats } from '../pages/chats/chats';
import { Profile } from '../pages/profile/profile';
import { Allposts } from '../pages/allposts/allposts';
import { Groups } from '../pages/groups/groups';
import { Buddies } from '../pages/buddies/buddies';
import { Passwordreset } from '../pages/passwordreset/passwordreset';
import { Imghandler } from '../providers/imghandler/imghandler';
import { User } from '../providers/user/user';
import { Phonecall } from '../pages/phonecall/phonecall';
import { Tabs2 } from '../pages/tabs2/tabs2';
import { Weatherhome } from '../pages/weatherhome/weatherhome';
import { Tabs } from '../pages/tabs/tabs';
import { Profiletab } from '../pages/profiletab/profiletab';
import { Groupstab } from '../pages/groupstab/groupstab';
import { Posttab } from '../pages/posttab/posttab';
import { Searchbuddiestab } from '../pages/searchbuddiestab/searchbuddiestab';
import { Allpoststab } from '../pages/allpoststab/allpoststab';
import { timer } from 'rxjs/observable/timer';
```

```
@Component({
  templateUrl: 'app.html'
})
export class MyApp {
  @ViewChild('content') nav: NavController
  rootPage:any = Login;
  avatar: any;
  displayName: string;
  mobile: string;
  pages:Array<{title:string, component:any}>;
  showSplash = true;
  constructor(platform: Platform,statusBar: StatusBar, splashScreen: SplashScreen,public userservice: User, public zone: NgZone, public alertCtrl: AlertController,
    public imghandler: Imghandler) {
    platform.ready().then(() => {
      // Okay, so the platform is ready and our plugins are available.
      // Here you can do any higher level native things you might need.
      statusBar.styleDefault();
      splashScreen.hide();
      timer(2000).subscribe(()=>this.showSplash=false)
    });
    this.pages=[
        { title:'Home',component: Allpoststab },
        { title:'Chat',component: Tabs },
        { title:'Groups',component: Groupstab },
        { title:'Profile',component: Profiletab },
```

```
      { title:'SearchBuddies',component: Searchbuddiestab },
      { title:'Send Post',component: Posttab },
      { title:'Music',component: Allposts },
      { title:'Weather',component: Tabs2 },
      { title:'Make A Call',component: Phonecall },
      { title:'Change Password',component: Passwordreset },
      { title:'Logout',component: Login }
    ];
  }
  openPage(page){
    if(page==='Logout'){
      firebase.auth().signOut().then(() => {
        this.nav.parent.parent.setRoot(Login);
      })
    }
    else{
       this.nav.setRoot(page.component);
    }  }
}
```

app.firebaseconfig.ts

```
// Initialize Firebase
 export var config = {
    apiKey: "AIzaSyDLbnz05z8U3Ll4D9-Zfifb-xU5vuqBhZA",
    authDomain: "todo-4469d.firebaseapp.com",
    databaseURL: "https://todo-4469d.firebaseio.com",
    projectId: "todo-4469d",
    storageBucket: "todo-4469d.appspot.com",
```

```
  messagingSenderId: "396313272057"
};
```

app.html

```
<div *ngIf="showSplash" class="splash">
  <div class="sk-cube-grid">
    <div class="sk-cube sk-cube1"></div>
    <div class="sk-cube sk-cube2"></div>
    <div class="sk-cube sk-cube3"></div>
    <div class="sk-cube sk-cube4"></div>
    <div class="sk-cube sk-cube5"></div>
    <div class="sk-cube sk-cube6"></div>
    <div class="sk-cube sk-cube7"></div>
    <div class="sk-cube sk-cube8"></div>
    <div class="sk-cube sk-cube9"></div>
  </div>
</div>
<ion-menu [content]="content">
    <ion-content>
        <ion-list>
            <ion-item>
                <div class="profile-image" (click)="editimage()">
                    <img src="{{avatar}}">
                </div>
                <div>
                    <h2 (click)="editname()">{{displayName}}</h2>
                </div>
                <div>
```

```
                    <h2 (click)="editmobile()">{{mobile}}</h2>
                  </div>
                  <div>
                    Tap on your pic or nick name to change it.
                  </div>
                  <div class="spacer" style="height: 10px;"></div>
                </ion-item>
              <button menuClose ion-item *ngFor="let p of pages" (click)="openPage(p)">
                {{p.title}}
              </button>
            </ion-list>
          </ion-content>
</ion-menu>
<ion-nav id="nav" #content [root]="rootPage"></ion-nav>
```

app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { ErrorHandler, NgModule } from '@angular/core';
import { IonicApp, IonicErrorHandler, IonicModule } from 'ionic-angular';
import { SplashScreen } from '@ionic-native/splash-screen';
import { StatusBar } from '@ionic-native/status-bar';
import { CallNumber } from '@ionic-native/call-number';
import { config } from './app.firebaseconfig';
import { AngularFireAuth } from 'angularfire2/auth';
import { AngularFireModule } from 'angularfire2';
import { File } from '@ionic-native/file';
```

```
import { FileChooser } from '@ionic-native/file-chooser';
import { FilePath } from '@ionic-native/file-path';
import { MyApp } from './app.component';
import { Login } from '../pages/login/login';
import { Tabs } from '../pages/tabs/tabs';
import { Auth } from '../providers/auth/auth';
import { User } from '../providers/user/user';
import { Chats } from '../pages/chats/chats';
import { Profile } from '../pages/profile/profile';
import { Groups } from '../pages/groups/groups';
import { Signup } from '../pages/signup/signup';
import { Profilepic } from '../pages/profilepic/profilepic';
import { Passwordreset } from '../pages/passwordreset/passwordreset';
import { Imghandler } from '../providers/imghandler/imghandler';
import { Buddies } from '../pages/buddies/buddies';
import { Requests } from '../providers/requests/requests';
import { Chat } from '../providers/chat/chat';
import { Buddychat } from '../pages/buddychat/buddychat';
import { Groupsprovider } from '../providers/groupsprovider/groupsprovider';
import { Newgroup } from '../pages/newgroup/newgroup';
import { Groupchat } from '../pages/groupchat/groupchat';
import { Groupbuddies } from '../pages/groupbuddies/groupbuddies';
import { Groupmembers } from '../pages/groupmembers/groupmembers';
import { Groupinfo } from '../pages/groupinfo/groupinfo';
import { Newsfeed } from '../pages/newsfeed/newsfeed';
import { Allposts } from '../pages/allposts/allposts';
import { Phonecall } from '../pages/phonecall/phonecall';
```

```
import { Tabs2 } from '../pages/tabs2/tabs2';
import { Weatherabout } from '../pages/weatherabout/weatherabout';
import { Weatherhome } from '../pages/weatherhome/weatherhome';
import { Weathersettings } from '../pages/weathersettings/weathersettings';
import { Weather } from '../providers/weather/weather';
import { HttpModule } from '@angular/http';
import { IonicStorageModule } from '@ionic/storage';
import { Profiletab } from '../pages/profiletab/profiletab';
import { Groupstab } from '../pages/groupstab/groupstab';
import { Allpoststab } from '../pages/allpoststab/allpoststab';
import { Searchbuddiestab } from '../pages/searchbuddiestab/searchbuddiestab';
import { Posttab } from '../pages/posttab/posttab';
import {BrowserAnimationsModule} from '@angular/platform-browser/animations'
import { Loginfail } from '../pages/loginfail/loginfail';

@NgModule({
  declarations: [
    MyApp, Login, Tabs, Chats, Profile, Groups, Signup, Profilepic, Passwordreset , Buddies, Buddychat, Newgroup, Groupchat, Groupbuddies, Groupmembers, Groupinfo, Newsfeed,
    Allposts, Phonecall, Tabs2, Weatherabout, Weatherhome, Weathersettings, Profiletab,
    Groupstab, Allpoststab, Searchbuddiestab, Posttab, Loginfail
  ],
  imports: [
    BrowserModule,
    BrowserAnimationsModule,
    HttpModule,
```

```
    IonicModule.forRoot(MyApp,{tabsPlacement:'top'}),
    AngularFireModule.initializeApp(config),
    IonicStorageModule.forRoot()
  ],
  bootstrap: [IonicApp],
  entryComponents: [
    MyApp, Login, Tabs, Chats, Profile, Groups, Signup, Profilepic, Passwordreset , Buddies, Buddychat, Newgroup, Groupchat, Groupbuddies, Groupmembers, Groupinfo, Newsfeed,
    Allposts, Phonecall, Tabs2, Weatherabout, Weatherhome, Weathersettings, Profiletab,
    Groupstab, Allpoststab, Searchbuddiestab, Posttab, Loginfail
  ],
  providers: [
    StatusBar,
    SplashScreen,
     File,
    FileChooser,
    FilePath,
    {provide: ErrorHandler, useClass: IonicErrorHandler},
   Auth,
   User,
   Chat,
   Groupsprovider,
   Requests,
   Imghandler,
   CallNumber,
   Weather,
```

```
    AngularFireAuth
  ]
})
export class AppModule {}
```

provider

auth.ts

```
import { Injectable } from '@angular/core';
import { AngularFireAuth } from 'angularfire2/auth';
import { usercreds } from '../../models/interfaces/usercred';
import { AlertController } from 'ionic-angular';
@Injectable()
export class Auth {
  constructor(public afireauth: AngularFireAuth,public alertCtrl: AlertController) {
    console.log('Hello Auth Provider');
  }
login(credentials: usercreds) {
  var promise = new Promise((resolve, reject) => {
    this.afireauth.auth.signInWithEmailAndPassword(credentials.email,
credentials.password).then(() => {
      resolve(true);
    }).catch((err) => {
      //reject(err);
      let alert = this.alertCtrl.create({
        title: 'Wrong Credentials',
        subTitle: err,
```

```
      buttons: ['Dismiss']
    });
    alert.present();
  })
 })
 return promise;
} }
```

chat.ts

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import firebase from 'firebase';
import { Events } from 'ionic-angular';
@Injectable()
export class Chat {
 firebuddychats = firebase.database().ref('/buddychats');
 buddy: any;
 buddymessages = [];
 constructor(public events: Events) {
 }
 initializebuddy(buddy) {
  this.buddy = buddy;
 }
 addnewmessage(msg) {
  if (this.buddy) {
   var promise = new Promise((resolve, reject) => {
    this.firebuddychats.child(firebase.auth().currentUser.uid).child(this.buddy.uid).push({
     sentby: firebase.auth().currentUser.uid,
```

```
        message: msg,
        timestamp: firebase.database.ServerValue.TIMESTAMP
      }).then(() => {
        this.firebuddychats.child(this.buddy.uid).child(firebase.auth().currentUser.uid).push({
          sentby: firebase.auth().currentUser.uid,
          message: msg,
          timestamp: firebase.database.ServerValue.TIMESTAMP
        }).then(() => {
          resolve(true);
          })
      })
    })
    return promise;
  }
}
getbuddymessages() {
  let temp;
  this.firebuddychats.child(firebase.auth().currentUser.uid).child(this.buddy.uid).on('value',
(snapshot) => {
    this.buddymessages = [];
    temp = snapshot.val();
    for (var tempkey in temp) {
      this.buddymessages.push(temp[tempkey]);
    }
    this.events.publish('newmessage');
  })
}
```

```
}
```

groupsproviders.ts

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Events } from 'ionic-angular';
import firebase from 'firebase';

@Injectable()
export class Groupsprovider {
  firegroup = firebase.database().ref('/groups');
  mygroups: Array<any> = [];
  currentgroup: Array<any> = [];
  currentgroupname;
  grouppic;
groupmsgs;
  constructor(public events: Events) {
  }
  addgroup(newGroup) {
    var promise = new Promise((resolve, reject) => {
      this.firegroup.child(firebase.auth().currentUser.uid).child(newGroup.groupName).set({
        groupimage: newGroup.groupPic,
        msgboard: '',
        owner: firebase.auth().currentUser.uid
      }).then(() => {
        resolve(true);
        }).catch((err) => {
```

```
        reject(err);
      })
    });
    return promise;
  }
  getmygroups() {
    this.firegroup.child(firebase.auth().currentUser.uid).once('value', (snapshot) => {
      this.mygroups = [];
      if(snapshot.val() != null) {
        var temp = snapshot.val();
        for (var key in temp) {
          var newgroup = {
            groupName: key,
            groupimage: temp[key].groupimage
          }
          this.mygroups.push(newgroup);
        }
      }
      this.events.publish('allmygroups');
    })
  }
  getintogroup(groupname) {
    if (groupname != null) {
      this.firegroup.child(firebase.auth().currentUser.uid).child(groupname).once('value',
(snapshot) => {
        if (snapshot.val() != null) {
          var temp = snapshot.val().members;
```

```
        this.currentgroup = [];
        for (var key in temp) {
          this.currentgroup.push(temp[key]);
        }
        this.currentgroupname = groupname;
        this.events.publish('gotintogroup');
      }
    })
  }
}
getownership(groupname) {
  var promise = new Promise((resolve, reject) => {
    this.firegroup.child(firebase.auth().currentUser.uid).child(groupname).once('value',
(snapshot) => {
      var temp = snapshot.val().owner;
      if (temp == firebase.auth().currentUser.uid) {
        resolve(true);
      }
      else {
        resolve(false);
      }
    }).catch((err) => {
      reject(err);
    })
  })
  return promise;
}
```

```
  getgroupimage() {
    return new Promise((resolve, reject) => {

this.firegroup.child(firebase.auth().currentUser.uid).child(this.currentgroupname).once('value'
, (snapshot) => {
        this.grouppic = snapshot.val().groupimage;
        resolve(true);
    })
    })
  }

  addmember(newmember) {

this.firegroup.child(firebase.auth().currentUser.uid).child(this.currentgroupname).child('mem
bers').push(newmember).then(() => {
      this.getgroupimage().then(() => {
        this.firegroup.child(newmember.uid).child(this.currentgroupname).set({
          groupimage: this.grouppic,
          owner: firebase.auth().currentUser.uid,
          msgboard: ''
        }).catch((err) => {
          console.log(err);
        })
      })
      this.getintogroup(this.currentgroupname);
    })
  }
  deletemember(member) {
```

```
  this.firegroup.child(firebase.auth().currentUser.uid).child(this.currentgroupname)
    .child('members').orderByChild('uid').equalTo(member.uid).once('value', (snapshot) => {
      snapshot.ref.remove().then(() => {
        this.firegroup.child(member.uid).child(this.currentgroupname).remove().then(() => {
          this.getintogroup(this.currentgroupname);
        })
      })
    })
  }

  getgroupmembers() {

this.firegroup.child(firebase.auth().currentUser.uid).child(this.currentgroupname).once('value'
, (snapshot) => {
    var tempdata = snapshot.val().owner;

this.firegroup.child(tempdata).child(this.currentgroupname).child('members').once('value',
(snapshot) => {
      var tempvar = snapshot.val();
      for (var key in tempvar) {
        this.currentgroup.push(tempvar[key]);
      }
    })
  })
  this.events.publish('gotmembers');
  }

  leavegroup() {
```

```
    return new Promise((resolve, reject) => {

this.firegroup.child(firebase.auth().currentUser.uid).child(this.currentgroupname).once('value'
, (snapshot) => {
      var tempowner = snapshot.val().owner;

this.firegroup.child(tempowner).child(this.currentgroupname).child('members').orderByChild
('uid')
        .equalTo(firebase.auth().currentUser.uid).once('value', (snapshot) => {
          snapshot.ref.remove().then(() => {

this.firegroup.child(firebase.auth().currentUser.uid).child(this.currentgroupname).remove().th
en(() => {
              resolve(true);
            }).catch((err) => {
              reject(err);
            })
          }).catch((err) => {
            reject(err);
          })
      })
    })
    })
  }


  deletegroup() {
    return new Promise((resolve, reject) => {

this.firegroup.child(firebase.auth().currentUser.uid).child(this.currentgroupname).child('mem
bers').once('value', (snapshot) => {
```

```
        var tempmembers = snapshot.val();


        for (var key in tempmembers) {
          this.firegroup.child(tempmembers[key].uid).child(this.currentgroupname).remove();
        }



this.firegroup.child(firebase.auth().currentUser.uid).child(this.currentgroupname).remove().then(() => {
          resolve(true);
        }).catch((err) => {
          reject(err);
        })


      })
    })
  }


  addgroupmsg(newmessage) {
    return new Promise((resolve) => {



this.firegroup.child(firebase.auth().currentUser.uid).child(this.currentgroupname).child('owner').once('value', (snapshot) => {
      var tempowner = snapshot.val();


this.firegroup.child(firebase.auth().currentUser.uid).child(this.currentgroupname).child('msgboard').push({
        sentby: firebase.auth().currentUser.uid,
```

```
      displayName: firebase.auth().currentUser.displayName,
      photoURL: firebase.auth().currentUser.photoURL,
      message: newmessage,
      timestamp: firebase.database.ServerValue.TIMESTAMP
    }).then(() => {
      if (tempowner != firebase.auth().currentUser.uid) {
        this.firegroup.child(tempowner).child(this.currentgroupname).child('msgboard').push({
          sentby: firebase.auth().currentUser.uid,
          displayName: firebase.auth().currentUser.displayName,
          photoURL: firebase.auth().currentUser.photoURL,
          message: newmessage,
          timestamp: firebase.database.ServerValue.TIMESTAMP
        })
      }
      var tempmembers = [];

this.firegroup.child(tempowner).child(this.currentgroupname).child('members').once('value',
(snapshot) => {
        var tempmembersobj = snapshot.val();
        for (var key in tempmembersobj)
          tempmembers.push(tempmembersobj[key]);
      }).then(() => {
        let postedmsgs = tempmembers.map((item) => {
          if (item.uid != firebase.auth().currentUser.uid) {
            return new Promise((resolve) => {
              this.postmsgs(item, newmessage, resolve);
            })
          }
```

```
        })
        Promise.all(postedmsgs).then(() => {
          this.getgroupmsgs(this.currentgroupname);
          resolve(true);
        })
       })
     })
   })
  }

  postmsgs(member, msg, cb) {
    this.firegroup.child(member.uid).child(this.currentgroupname).child('msgboard').push({
          sentby: firebase.auth().currentUser.uid,
          displayName: firebase.auth().currentUser.displayName,
          photoURL: firebase.auth().currentUser.photoURL,
          message: msg,
          timestamp: firebase.database.ServerValue.TIMESTAMP
    }).then(() => {
      cb();
    })
  }

  getgroupmsgs(groupname) {

this.firegroup.child(firebase.auth().currentUser.uid).child(groupname).child('msgboard').on('value', (snapshot) => {
      var tempmsgholder = snapshot.val();
```

```
      this.groupmsgs = [];
      for (var key in tempmsgholder)
        this.groupmsgs.push(tempmsgholder[key]);
      this.events.publish('newgroupmsg');
    })
  }
}
```

imagehandler.ts

```
import { Injectable } from '@angular/core';
import { FileChooser } from '@ionic-native/file-chooser';
import { File } from '@ionic-native/file';
import { AngularFireAuth } from 'angularfire2/auth';
import { FilePath } from '@ionic-native/file-path';
import * as firebase from 'firebase';

@Injectable()
export class Imghandler {
  afireauth: any;
  firestore = firebase.storage();
  firedata = firebase.database().ref('/newposts');
  nativepath: any;
  constructor(public filechooser: FileChooser) {
  }
  uploadimage() {
    var promise = new Promise((resolve, reject) => {
        this.filechooser.open().then((url) => {
```

```
        (<any>window).FilePath.resolveNativePath(url, (result) => {
          this.nativepath = result;
          (<any>window).resolveLocalFileSystemURL(this.nativepath, (res) => {
            res.file((resFile) => {
              var reader = new FileReader();
              reader.readAsArrayBuffer(resFile);
              reader.onloadend = (evt: any) => {
                var imgBlob = new Blob([evt.target.result], { type: 'image/jpeg' });
                var imageStore = this.firestore.ref('/profileimages').child(firebase.auth().currentUser.uid);
                imageStore.put(imgBlob).then((res) => {

this.firestore.ref('/profileimages').child(firebase.auth().currentUser.uid).getDownloadURL().then((url) => {
                    resolve(url);
                  }).catch((err) => {
                     reject(err);
                  })
                }).catch((err) => {
                  reject(err);
                })
              }
            })
          })
        })
      })
    })
    return promise;
```

```
  }
  picmsgstore() {
    var promise = new Promise((resolve, reject) => {
      this.filechooser.open().then((url) => {
        (<any>window).FilePath.resolveNativePath(url, (result) => {
          this.nativepath = result;
          (<any>window).resolveLocalFileSystemURL(this.nativepath, (res) => {
            res.file((resFile) => {
              var reader = new FileReader();
              reader.readAsArrayBuffer(resFile);
              reader.onloadend = (evt: any) => {
                var imgBlob = new Blob([evt.target.result], { type: 'image/jpeg' });
                var uuid = this.guid();
                var imageStore = this.firestore.ref('/picmsgs').child(firebase.auth().currentUser.uid).child('picmsg' + uuid);
                imageStore.put(imgBlob).then((res) => {
                  resolve(res.downloadURL);
                }).catch((err) => {
                  reject(err);
                })
              .catch((err) => {
                reject(err);
              })
            }
          })
        })
      })
```

```
    })
  })
   return promise;
 }

 guid() {
 function s4() {
  return Math.floor((1 + Math.random()) * 0x10000)
    .toString(16)
    .substring(1);
 }
 return s4() + s4() + '-' + s4() + '-' + s4() + '-' +
  s4() + '-' + s4() + s4() + s4();
}

grouppicstore(groupname) {
 var promise = new Promise((resolve, reject) => {
    this.filechooser.open().then((url) => {
      (<any>window).FilePath.resolveNativePath(url, (result) => {
        this.nativepath = result;
        (<any>window).resolveLocalFileSystemURL(this.nativepath, (res) => {
          res.file((resFile) => {
            var reader = new FileReader();
            reader.readAsArrayBuffer(resFile);
            reader.onloadend = (evt: any) => {
              var imgBlob = new Blob([evt.target.result], { type: 'image/jpeg' });
```

```
            var imageStore = this.firestore.ref('/groupimages').child(firebase.auth().currentUser.uid).child(groupname);
            imageStore.put(imgBlob).then((res) => {

this.firestore.ref('/profileimages').child(firebase.auth().currentUser.uid).child(groupname).getDownloadURL().then((url) => {
                resolve(url);
              }).catch((err) => {
                reject(err);
              })
            }).catch((err) => {
              reject(err);
            })
          }
        })
      })
    })
  })
   return promise;
}
uploadimagepost() {
  var promise = new Promise((resolve, reject) => {
     this.filechooser.open().then((url) => {
       (<any>window).FilePath.resolveNativePath(url, (result) => {
         this.nativepath = result;
         (<any>window).resolveLocalFileSystemURL(this.nativepath, (res) => {
           res.file((resFile) => {
```

```
            var reader = new FileReader();
            reader.readAsArrayBuffer(resFile);
            reader.onloadend = (evt: any) => {
              var imgBlob = new Blob([evt.target.result], { type: 'image/jpeg' });
              var uuid = this.guid();
              var imageStore = this.firestore.ref('/postingpic').child(firebase.auth().currentUser.uid).child('postingpic' + uuid);
              imageStore.put(imgBlob).then((res) => {

this.firestore.ref('/postingpic').child(firebase.auth().currentUser.uid).child('postingpic' + uuid).getDownloadURL().then((url) => {
                resolve(url);
              }).catch((err) => {
                reject(err);
              })
            }).catch((err) => {
              reject(err);
            })
          }
        })
      })
    })
  })
   return promise;
}
updateimagepost(imageurl) {
  var promise = new Promise((resolve, reject) => {  //actual
```

```
    this.afireauth.auth.currentUser.updateProfile({
      displayName: this.afireauth.auth.currentUser.displayName,
      photoURL: imageurl
    }).then(() => {
     var uuid = this.guid();//actual
     this.firestore.ref('/postingpic').child(firebase.auth().currentUser.uid).child('postingpic'  +
uuid);//actual
      firebase.database().ref('/newposts/' + firebase.auth().currentUser.uid).update({
      displayName: this.afireauth.auth.currentUser.displayName,
      photoURL: imageurl,
      uid: firebase.auth().currentUser.uid
      }).then(() => {
        resolve({ success: true });  //actual
        }).catch((err) => {
          reject(err);
        })
    }).catch((err) => {
        reject(err);
      })
  })
  return promise;
} }
```

requests.ts

```
import { Injectable } from '@angular/core';
import { connreq } from '../../models/interfaces/request';
import firebase from 'firebase';
import { Events } from 'ionic-angular';
```

```
import { User } from '../user/user';
@Injectable()
export class Requests {
  firereq = firebase.database().ref('/requests');
  firefriends = firebase.database().ref('/friends');

  userdetails;
  myfriends;
  constructor(public userservice: User, public events: Events) {
  }
  getmyfriends() {
    let friendsuid = [];
   this.firefriends.child(firebase.auth().currentUser.uid).on('value', (snapshot) => {
     let allfriends = snapshot.val();
     this.myfriends = [];
     for (var i in allfriends)
       friendsuid.push(allfriends[i].uid);
     this.userservice.getallusers().then((users) => {
       this.myfriends = [];
       for (var j in friendsuid)
         for (var key in users) {
           if (friendsuid[j] === users[key].uid) {
             this.myfriends.push(users[key]);
           }
         }
       this.events.publish('friends');
     }).catch((err) => {
```

```
      alert(err);
    })
  })
}
sendrequest(req: connreq) {
  var promise = new Promise((resolve, reject) => {
    this.firereq.child(req.recipient).push({
    sender: req.sender
    }).then(() => {
      resolve({ success: true });
      }).then((err) => {
       // resolve(err);
  })
  })
  return promise;
}
getmyrequests() {
  let allmyrequests;
  var myrequests = [];
  this.firereq.child(firebase.auth().currentUser.uid).on('value', (snapshot) => {
    allmyrequests = snapshot.val();
    myrequests = [];
    for (var i in allmyrequests) {
      myrequests.push(allmyrequests[i].sender);
    }
    this.userservice.getallusers().then((res) => {
      var allusers = res;
```

```
        this.userdetails = [];
        for (var j in myrequests)
          for (var key in allusers) {
            if (myrequests[j] === allusers[key].uid) {
              this.userdetails.push(allusers[key]);
            }
          }
        this.events.publish('gotrequests');
      })

  })
}
acceptrequest(buddy) {
  var myfriends = [];
  var promise = new Promise((resolve, reject) => {
    this.firefriends.child(firebase.auth().currentUser.uid).push({
      uid: buddy.uid
    }).then(() => {
        this.deleterequest(buddy).then(() => {
        resolve(true);
      })

     /* })/*.catch((err) => {
        reject(err);
       })*/
      })
  })
```

```
  return promise;
}
deleterequest(buddy) {
  var promise = new Promise((resolve, reject) => {
this.firereq.child(firebase.auth().currentUser.uid).orderByChild('sender').equalTo(buddy.uid).
once('value', (snapshot) => {
        let somekey;
        for (var key in snapshot.val())
          somekey = key;
        this.firereq.child(firebase.auth().currentUser.uid).child(somekey).remove().then(() => {
          resolve(true);
        })
       })
        .then(() => {

      }).catch((err) => {
        reject(err);
      })
  })
  return promise;
}
}
```

users.ts

```
import { Injectable } from '@angular/core';
import { AngularFireAuth } from 'angularfire2/auth';
import firebase, { firestore, database } from 'firebase';
import { Events, AlertController } from 'ionic-angular';
```

```
@Injectable()
export class User {
  bun: any;
  uiddata:any;
  buddyposts = [];
  firestore = firebase.storage();
  firedata = firebase.database().ref('/chatusers');
  firedata1 = firebase.database().ref('/newposts');
   constructor(public afireauth: AngularFireAuth,public events: Events,public alertCtrl: AlertController) {
  }
  adduser(newuser) {
    var promise = new Promise((resolve, reject) => {
      this.afireauth.auth.createUserWithEmailAndPassword(newuser.email, newuser.password).then(() => {
        this.afireauth.auth.currentUser.updateProfile({
          displayName: newuser.displayName,
          photoURL: 'https://firebasestorage.googleapis.com/v0/b/myapp-4eadd.appspot.com/o/chatterplace.png?alt=media&token=e51fa887-bfc6-48ff-87c6-e2c61976534e'
        }).then(() => {
          this.firedata.child(this.afireauth.auth.currentUser.uid).set({
            uid: this.afireauth.auth.currentUser.uid,
            displayName: newuser.displayName,
            mobile: newuser.mobile,
            photoURL: 'https://firebasestorage.googleapis.com/v0/b/myapp-4eadd.appspot.com/o/chatterplace.png?alt=media&token=e51fa887-bfc6-48ff-87c6-e2c61976534e'
          }).then(() => {
```

```
          resolve({ success: true });
          }).catch((err) => {
            reject(err);
        })
        }).catch((err) => {
          reject(err);
      })
    }).catch((err) => {
      //reject(err);
      let alert = this.alertCtrl.create({
        title: 'Details wrongly entered',
        subTitle: err,
        buttons: ['Dismiss']
      });
      alert.present();
    })
  })
  return promise;
}
passwordreset(email) {
  var promise = new Promise((resolve, reject) => {
    firebase.auth().sendPasswordResetEmail(email).then(() => {
      resolve({ success: true });
    }).catch((err) => {
      //reject(err);
      let alert = this.alertCtrl.create({
        title: 'Wrong Email',
```

```
        subTitle: err,
        buttons: ['Dismiss']
      });
      alert.present();
    })
  })

  return promise;
}

updateimage(imageurl) {
  var promise = new Promise((resolve, reject) => {
    this.afireauth.auth.currentUser.updateProfile({
      displayName: this.afireauth.auth.currentUser.displayName,
      photoURL: imageurl
    }).then(() => {
      firebase.database().ref('/users/' + firebase.auth().currentUser.uid).update({
      displayName: this.afireauth.auth.currentUser.displayName,
      photoURL: imageurl,
      uid: firebase.auth().currentUser.uid
      }).then(() => {
        resolve({ success: true });
        }).catch((err) => {
          reject(err);
        })
//for chatusers
        firebase.database().ref('/chatusers/' + firebase.auth().currentUser.uid).update({
```

```
            displayName: this.afireauth.auth.currentUser.displayName,
            photoURL: imageurl,

            uid: firebase.auth().currentUser.uid
            }).then(() => {
               resolve({ success: true });
               }).catch((err) => {
                  reject(err);
               })
       }).catch((err) => {
            reject(err);
         })
   })
   return promise;
}

updatedisplayname(newname) {
  var promise = new Promise((resolve, reject) => {
   this.afireauth.auth.currentUser.updateProfile({
   displayName: newname,
   photoURL: this.afireauth.auth.currentUser.photoURL
  }).then(() => {
   this.firedata.child(firebase.auth().currentUser.uid).update({
     displayName: newname,
     photoURL: this.afireauth.auth.currentUser.photoURL,
     uid: this.afireauth.auth.currentUser.uid
   }).then(() => {
```

```
      resolve({ success: true });
    }).catch((err) => {
      reject(err);
    })
    }).catch((err) => {
      reject(err);
  })
  })
  return promise;
}

updatemobilenumber(newname) {
  var promise = new Promise((resolve, reject) => {
    this.afireauth.auth.currentUser.updateProfile({
    displayName:this.afireauth.auth.currentUser.displayName,
    photoURL: this.afireauth.auth.currentUser.photoURL
  }).then(() => {
    this.firedata.child(firebase.auth().currentUser.uid).update({

      mobile:newname,
      photoURL: this.afireauth.auth.currentUser.photoURL,
      uid: this.afireauth.auth.currentUser.uid
    }).then(() => {
      resolve({ success: true });
    }).catch((err) => {
      reject(err);
    })
```

```
    }).catch((err) => {
      reject(err);
  })
  })
  return promise;
}



getuserdetails() {
  var promise = new Promise((resolve, reject) => {
  this.firedata.child(firebase.auth().currentUser.uid).once('value', (snapshot) => {
    resolve(snapshot.val());
  }).catch((err) => {
    reject(err);
    })
  })
  return promise;
}

getallusers() {
  var promise = new Promise((resolve, reject) => {
    this.firedata.orderByChild('uid').once('value', (snapshot) => {
      let userdata = snapshot.val();
      let temparr = [];
      for (var key in userdata) {
        temparr.push(userdata[key]);
      }
```

```
      resolve(temparr);
    }).catch((err) => {
      reject(err);
    })
  })
  return promise;
}
//for posting
/*updateimagepost(imageurl) {
  var promise = new Promise((resolve, reject) => {

       this.firestore.ref('/postingpic').child(firebase.auth().currentUser.uid).child('postingpic' +
uuid);
         //firebase.database().ref('/postingpic/' +
firebase.auth().currentUser.uid).child(firebase.auth().currentUser.uid)/*.update({

            resolve({ success: true });

  })
  return promise;
}*/

guid() {
  function s4() {
    return Math.floor((1 + Math.random()) * 0x10000)
      .toString(16)
      .substring(1);
  }
```

```
  return s4() + s4() + '-' + s4() + '-' + s4() + '-' +
    s4() + '-' + s4() + s4() + s4();
}

//for adding post
addpost(newpost,uuid){
  var promise = new Promise((resolve, reject) => {
    this.firedata1.push({
      uid: this.afireauth.auth.currentUser.uid,
      displayName:this.afireauth.auth.currentUser.displayName,
      posttitle: newpost.posttitle,
      postmessage:newpost.postmessage,
      profileURL:this.afireauth.auth.currentUser.photoURL,
      timestamp: firebase.database.ServerValue.TIMESTAMP,
      photoURL:  uuid,
    }).then(() => {
      resolve({ success: true });
      })//.catch((err) => {
       // reject(err);
    //})
})
  return promise;

}
//updating the image url method
updateimagepostingurl(imageurl) {
  var promise = new Promise((resolve, reject) => {
```

```
      this.afireauth.auth.currentUser.updateProfile({
         displayName: this.afireauth.auth.currentUser.displayName,
         photoURL: imageurl
      }).then(() => {
         firebase.database().ref('/users/' + firebase.auth().currentUser.uid).update({

         displayName: this.afireauth.auth.currentUser.displayName,
         photoURL: imageurl,
         uid: firebase.auth().currentUser.uid
         }).then(() => {
            resolve({ success: true });
            }).catch((err) => {
               reject(err);
            })
      }).catch((err) => {
           reject(err);
        })
  })
  return promise;
}
//get the details of posts of all users
getdetailsofallposts(){
  var promise = new Promise((resolve, reject) => {
  let temp;
    this.firedata1.on('value', (snapshot) => {
      this.buddyposts = [];
      temp = snapshot.val();
```

```
      for (var tempkey in temp) {
        this.buddyposts.push(temp[tempkey]);
      }
      this.events.publish('newpostings');
    })
})
return promise;
}
}
```

weather.ts

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Http } from '@angular/http';
import 'rxjs/add/operator/map';
@Injectable()
export class Weather {
  apikey = '99dfe35fcb7de1ee';
  url;
    constructor(public http: Http) {
      console.log('Hello WeatherProvider Provider');
      this.url = 'http://api.wunderground.com/api/'+this.apikey+'/conditions/q';
  }
  getWeather(city, state){
    return this.http.get(this.url+'/'+state+'/'+city+'.json')
    .map(res => res.json());
```

```
  }
}
```

variable.scss

```
$font-path: "../assets/fonts";
$app-direction: ltr;
@import "ionic.globals";
$colors: (
  primary:    #488aff,
  secondary:  #32db64,
  danger:     #f53d3d,
  light:      #f4f4f4,
  dark:       #222,
 hcolor: #5682cf,// #d33257,//#5682cf,  //306cd4   5682cf
 hcolor1:#187ea7,
  //hcolor:     #03b6b3,
);
@import "ionic.theme.default";
@import "ionic.ionicons";
@import "roboto";
@import "noto-sans";
```