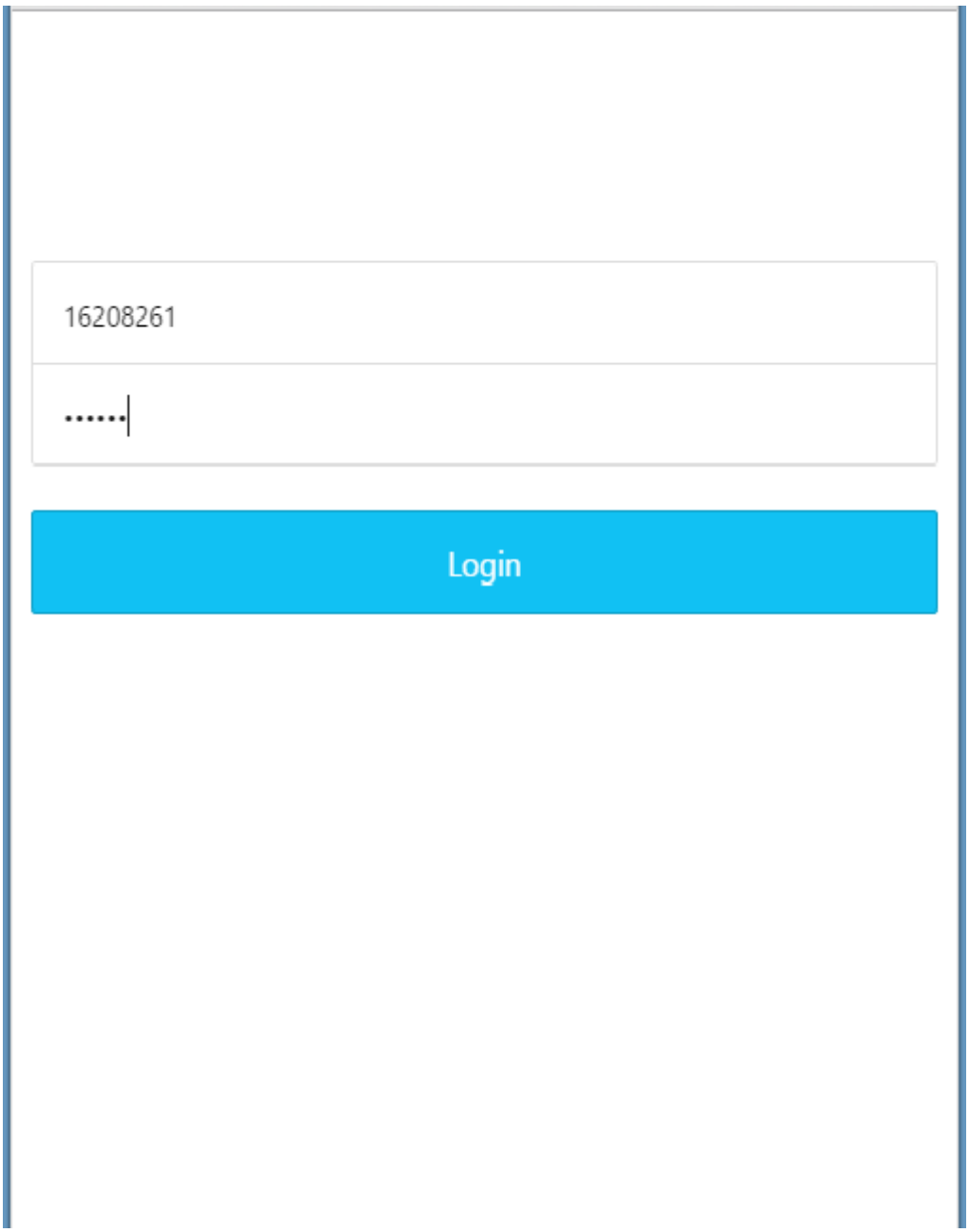
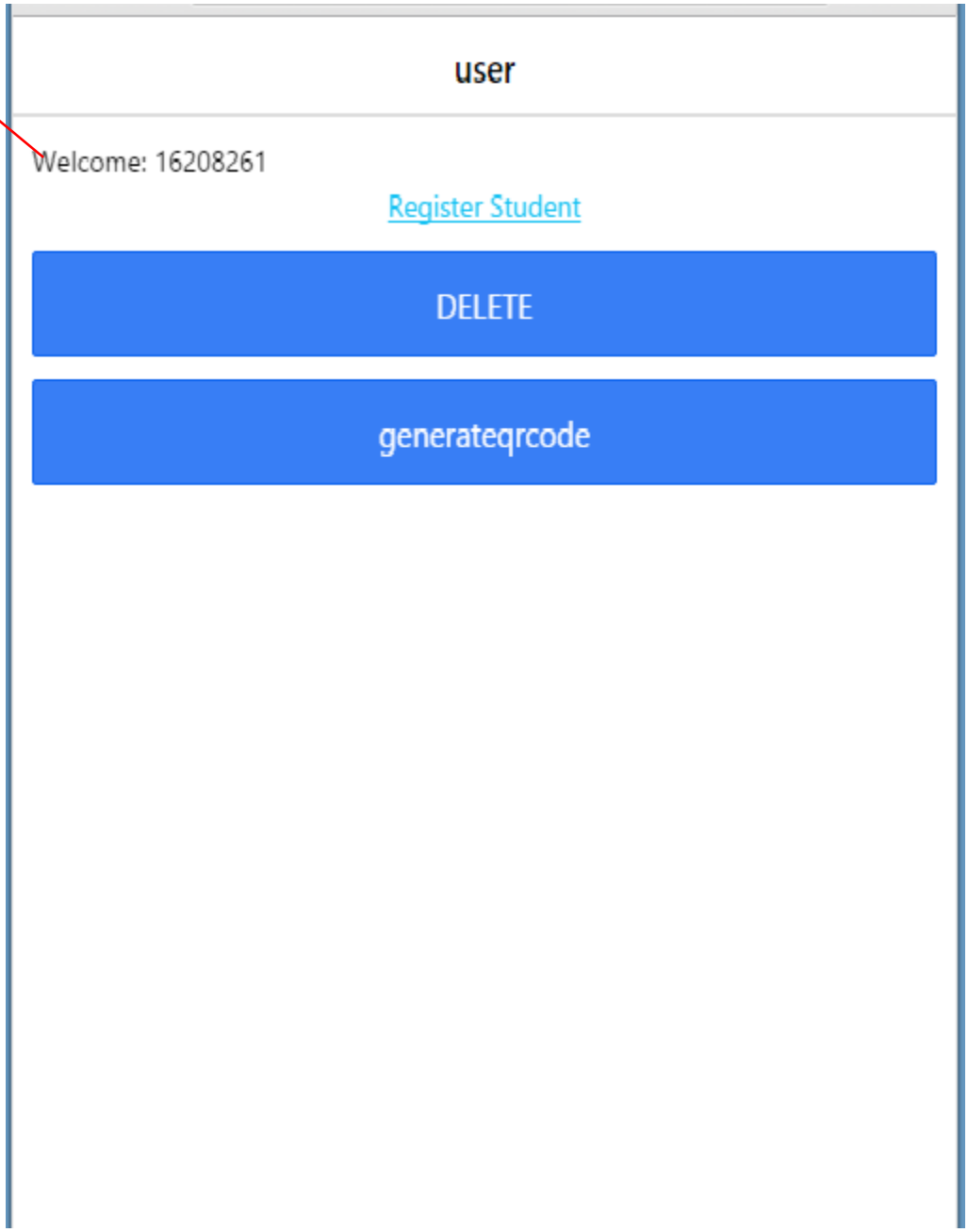
Login page



Singleton Pattern

After Logging in



Mixin Pattern

Singleton Pattern

```
var User = (function () {
  // Instance stores a reference to the Singleton
  var instance;

  function init() {
    // Singleton
    // Private methods and variables
    function privateMethod(){
        console.log( "I am private" );
    }

    var privateVariable = "Im also private";

    return {
      // Public methods
      username: function () {
        return instance.username;
      },
      //Public Properties
      username : "username"
    };
  };

  return {
    // Get the Singleton instance if one exists
    // or create one if it doesn't
    getInstance: function () {
      if ( !instance ) {
        instance = init();
      }
      return instance;
    }
```

```
.controller('adminCtrl', function($scope, $state) {
    var username = User.getInstance();

  console.log(username.username()); // true



  $scope.username = user.username;
    $scope.generateqrcode=function() {
        $state.go('qrpage');
    }
})
```

Mixin Pattern

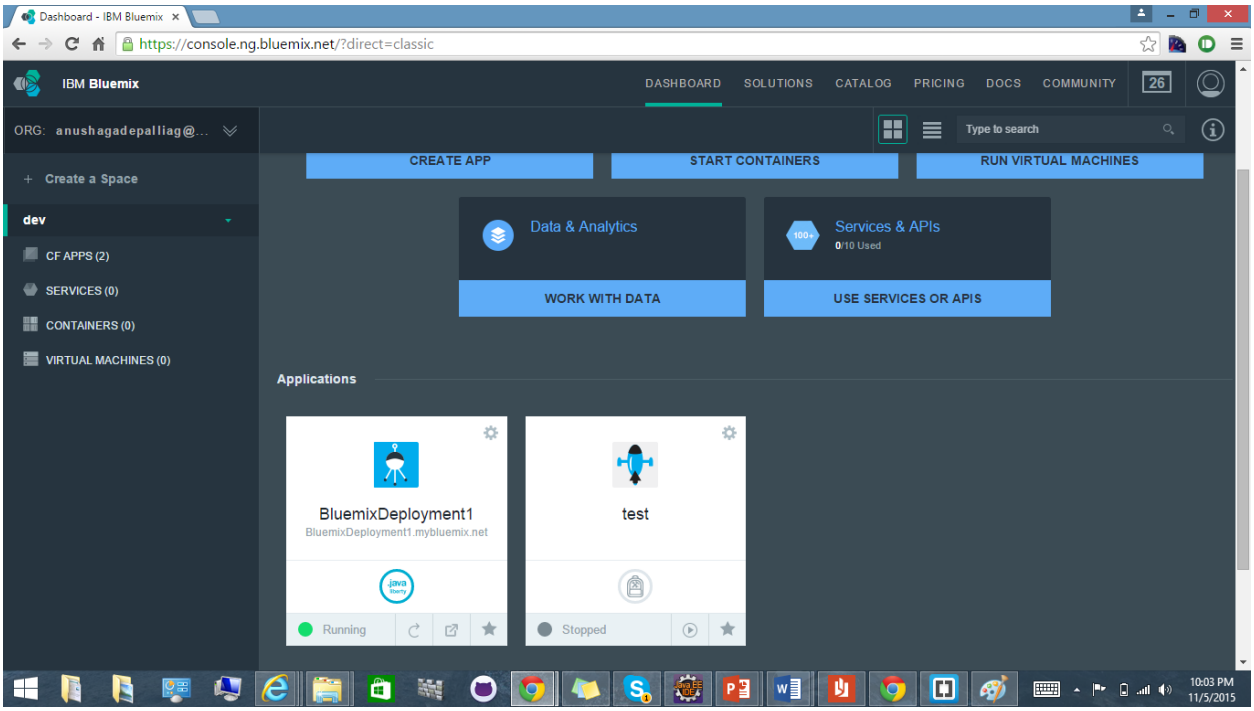
```
var Mixin  = function() {}
Mixin.prototype = {
  login: function(){
    console.log( "user logging in" );
  },
  delete: function(){
    console.log( "deleting a user account" );
  },
  register: function(){
    console.log( "user can register now" );
  }
};

// A skeleton carAnimator constructor
function Student() {
  this.login = function(){
    console.log( "student logs in" );
  };
}

// A skeleton personAnimator constructor
function Admin(){
  this.delete = function(){
    console.log("admin can delete student account");
    this.register = function(){
    console.log("admin can register a student");
  };
}

augment(Student, Mixin);
augment(Admin, Mixin);

```

```

.controller('LoginCtrl', function($scope, LoginService,UpdateService, DeleteService,  $ionicPopup, $state) {
    $scope.data = {};
  augment(Inside, Mixin);

 var inside = new Inside();

    $scope.login = function(username) {
        LoginService.loginUser($scope.data.username, $scope.data.password).success(function(data) {
            $state.go('adminhome');
        }).error(function(data) {
            var alertPopup = $ionicPopup.alert({
                title: 'Login failed!',
                template: 'Please check your credentials!'
            });
        });

    }
```

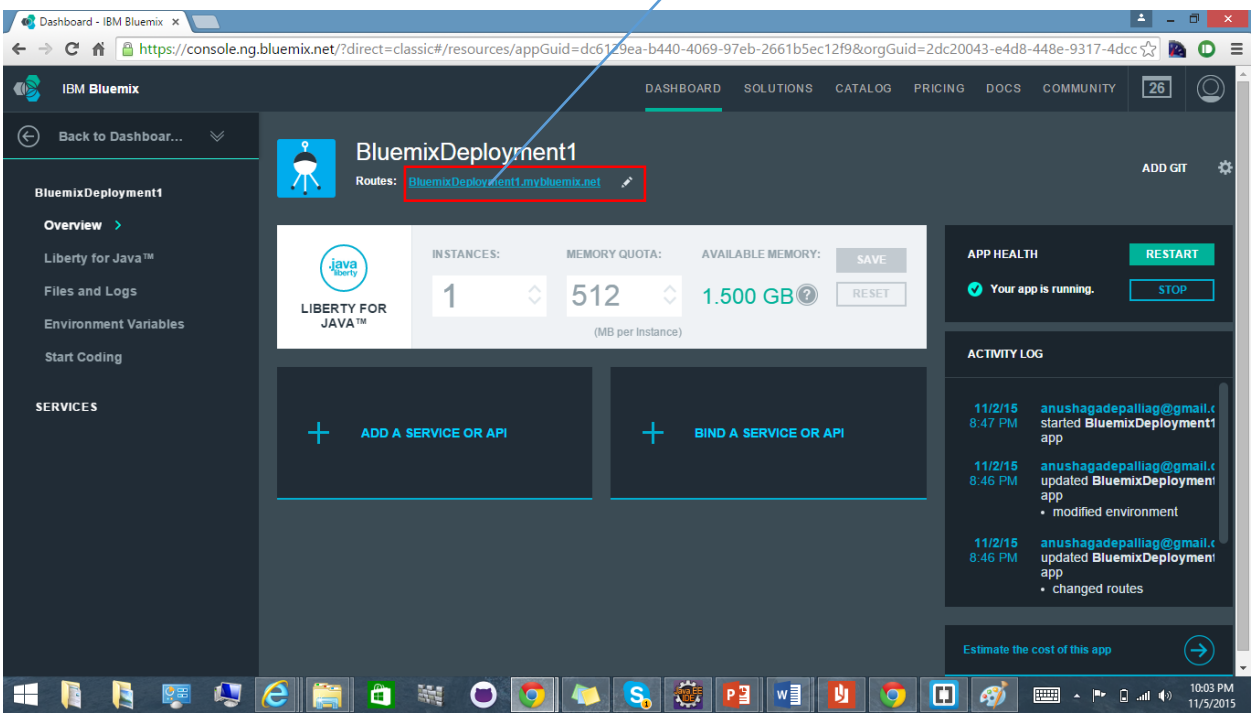
Factory Pattern

```
function StudentDatabase(options) {
  // some defaults
  this.idnumber = options.idnumber || localStorage.getItem("username");
  this.name = options.name || localStorage.getItem("fname");
  this.email = options.email || localStorage.getItem("email");
}

// A constructor for defining new trucks
/*function Truck(options) {
  this.state = options.state || "used";
  this.wheelSize = options.wheelSize || "large";
  this.color = options.color || "blue";
}*/

// Define a skeleton vehicle factory
function DatabaseFactory() {}
// Define the prototypes and utilities for this factory

// Our default vehicleClass is Car
DatabaseFactory.prototype.studentClass = StudentDatabase;

// Our Factory method for creating new Vehicle instances
DatabaseFactory.prototype.createStudent = function ( options ) {

  return new this.studentClass( options );
};

```

App deployment on bluemix



Application URL



Application running on bluemix