Student Info Management

# Summary

The application is used to manage students’ information. Each anonymous user can access the login and sing up page.

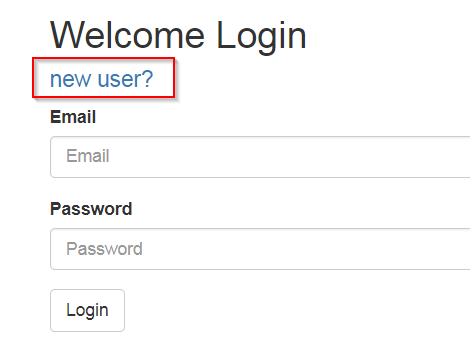
For registered user, after sing in will open his profile page in which he can update his information. For user has an “ADMIN” role can open the students list page where the admin can perform more options like remove a student.

The backend of the application using SpringBoot(Spring MVC, Spring Data JPA, Spring Security) and frontend using standard HTML and Java script in which using Ajax (XMLHttpRequest) to communicate with the backend. Data exchange format is JSON.

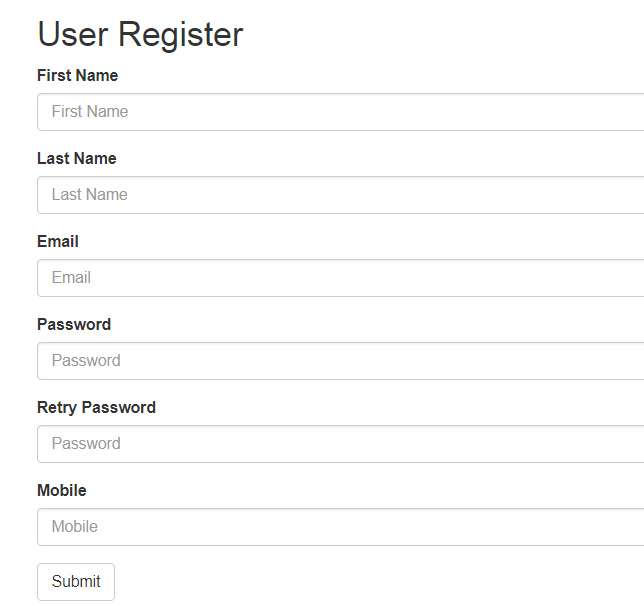
# Functional Features

The application meets the functional features required in the assessment detailed as below.

## 2.1 Sing up

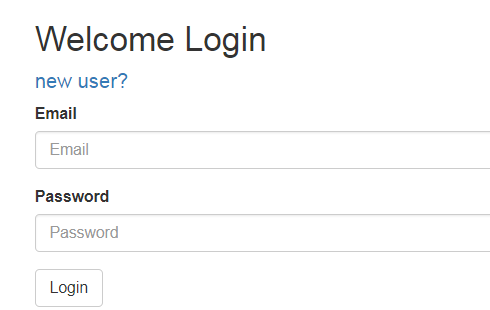


Click “new user?” on the index page to open the register page as below:



User should input all the fields because they are required, and also each field has validation.

## 2.2 login



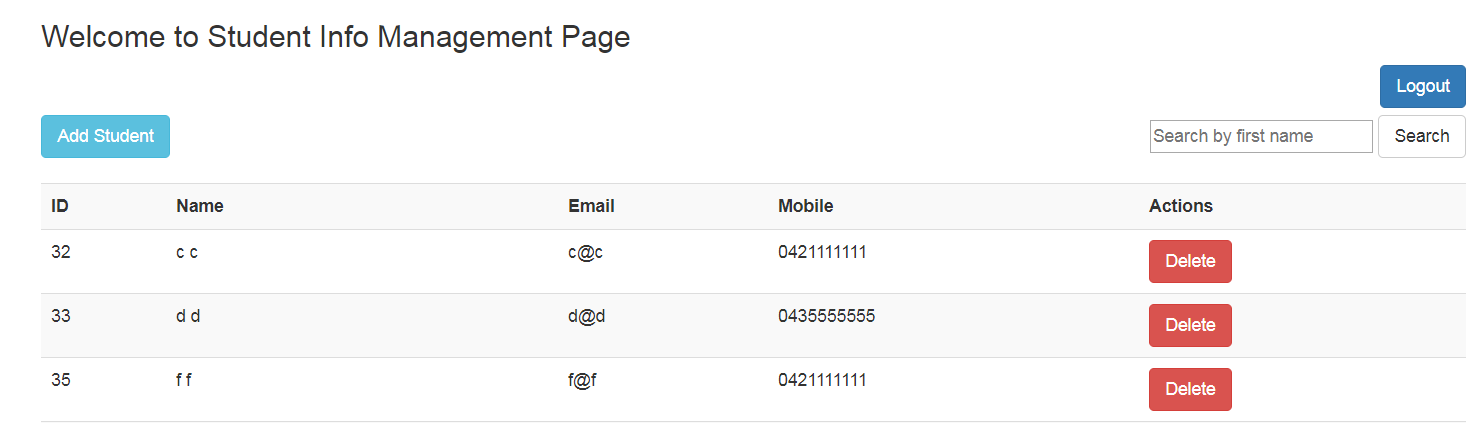
By input the email and password, the user can login. If it is a normal user and successfully login will direct to the profile page in which can update some of his information, if login as admin will direct to student list page. For normal user, the profile page is as below:



## 2.3 admin space and ability to manage data

For admin, the admin name is configured in property file and also persist in DB. Only the user has “ROLE\_ADMIN” has the access to student list and student add pages.

After admin login will direct to student list page as below:



On this page, the admin can delete a user or add new student.

## 2.4 logout

By clicking the Logout button on the top-right, will direct to the index page.

## 2.5 form validation

For the student add page, we add some basic validation as below:

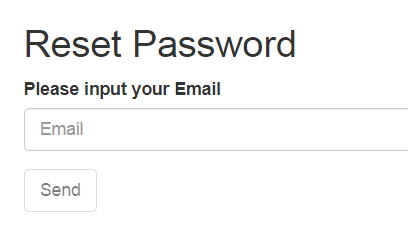
|  |  |
| --- | --- |
| **Input** | **Validation rule** |
| First name | Required |
| Last name | Required |
| Email | Required  Match Email format |
| Password | Required  minlength is 4  maxlength is 8  should be the same for two input passwords |
| Mobile | Required  Regular expression to match Australia mobile format |

If any of the above field is not valid, user will get alert.

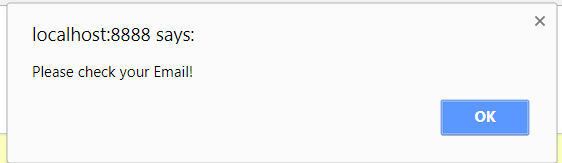
## 2.6 Email based password reset

If a registered user forget the password, can click the link to reset:

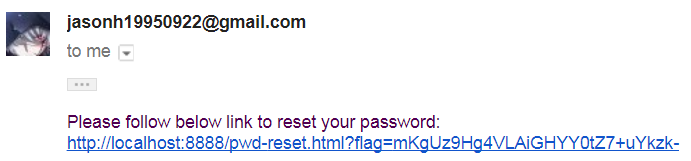
First step need to input the Email:



Only if the user input Email is valid (not empty, match Email pattern and is registered in our application) will the “Send” button enabled. From the implementation perspective, when the input on blur, a request will be sent to backend with the input Email and check the Database if it is an Email already registered in the application. If it is the button will be enable and use can click, after that will get an alert that a password reset message was sent to the input Email.



Open your Email box and will see some message like below:



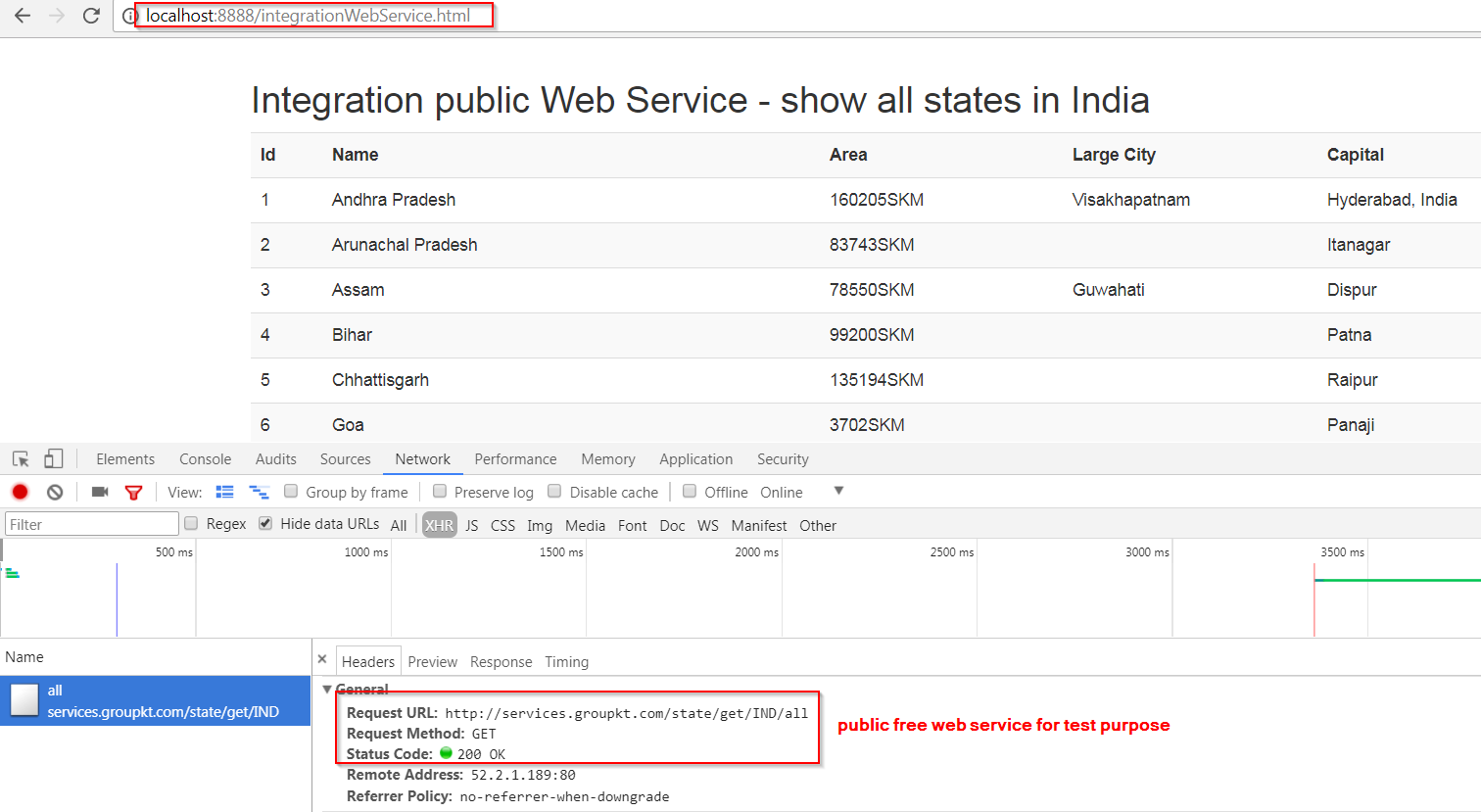
Click the link in the Email will open the password reset page as below:



Input the new password and press “Reset” button, if success will get a notice and redirect to login page. Then user can use the new password to login.

## 2.7 integration with public web service

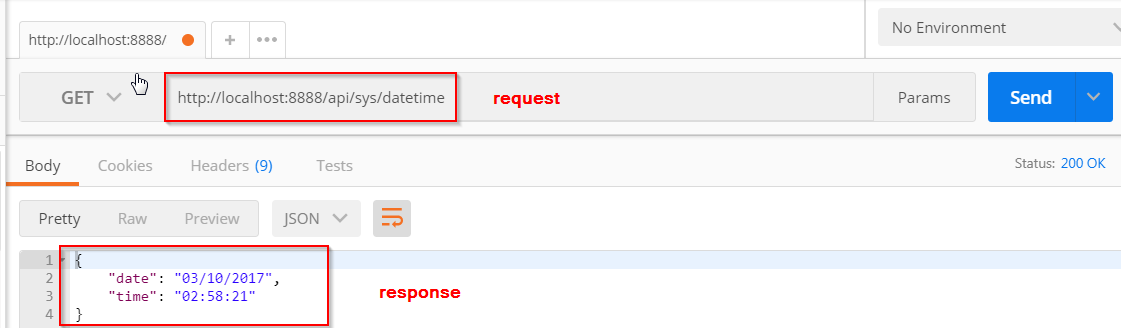
We have found a free web service getting all states of a country (India), click the link “Integration with public web service” in index page will show all states information in India as below:



## 2.8 expose a useful RESTful API

The application exposes a RESTful API to get the current system date and time:

Endpoint: /api/sys/datetime



## 2.9 deployment to cloud

Deployed in AWS (EC2 server and MySQL DB)

# Non-functional features

## 3.1 AJAX

Using XMLHttpRequest object to communicate with backend. The code is in app.js and some html files.

## 3.2 concurrent processing

The application can handle multiple user login and using the application.

## 3.3 modern framework

Using Spring technology including Spring Core, Spring MVC, Spring Data JPA, Spring IO, Spring Security.

## 3.4 Database

Using MySQL to persist data.

## 3.5 security

Using JWT(JSON Web Tokens which are an open, industry standard RFC 7519 method for representing claims securely between two parties.) and Spring security to enable both the frontend security and backend web security for APIs.

## 3.6 must not store unencrypted password

Password is encrypted in both frontend (SHA), and backend with BCrypt.

## 3.7 authentication framework

Using JWT and Spring Login web security.

## 3.8 documents and comments

Detailed documents with text and screenshots.

## 3.9 source control and code tracking

Deployed in GitHub for source control and tracking purpose.