# 7. System Validation

This chapter coves our system validation, implicating all the testing process of the system. This chapter is divided into two sections. Section 7.1 Subsystem Tests, which covers all the testing done for each subsystem in specific preceding the integration process. Section 7.2 System Tests, covering all the system tests directed after integrating the modules of the system.

|  |  |
| --- | --- |
| Subsystems groups | Test Description |
| Frontend UI (Client Subsystem) and backend Web Services | This tests that all the request generated on the frontend are interpreted in the backend webservice returning the result expected. This will only validate that the string to perform the request is correct and that the return is the json object projected. |
| Backend Web Services and backend Favorite Alarms | This tests that a favorite alarm can be added, removed, or set defined time to trigger through the Web Service, and that favorite alarm triggers a message. This Email message is sent to the user that registered that favorite alarm. |
| Backend Web Services and backend Email Service | This tests checks that Emails generated are being received by the correct recipients with the defined template and attachments if so. |

The system is comprised of four subsystems. Among these four subsystems there were three groups that needed to have subsystem testing done on them. These three groups are as follows, the frontend UI (client subsystem) and the backend Web services, the backend Web services and the backend Favorite Alarms, the backend Web services and backend Email service. We will now identify the subsystem testing that took place for each of these groups.

This project had very little subsystem testing that required to be completed. During the planning stage of designing our application we did it in a way that the exchange of information between subsystems were as minimal as possible. We decided to do this so that we would have very little connection between subsystems. In that way our system will become more consistent.

For the first subsystem group, we created the Web services taking to account that few information will be passed as parameters by the client subsystem to each of them and sent in the form of Post request. Json format was returned with minimum information possible. Continues testing was accomplished using browser Rest tools, given the fact that the frontend and backend interaction is only through web services requests. We conducted manual testing of the interaction of this method between the two subsystems

For the second subsystem group, we used the Web Service to make possible that a favorite alarm can be added, removed, or set defined time to trigger through the Web Service, and that favorite alarm triggers a message in the form of Post request. Json format was returned with minimum information possible stating the status of that favorite alarm. To check this first part the testing was accomplished using browser Rest tools like in the first subsystem group testing. The second part was checked that the Email message is sent to the user that registered that favorite alarm during the time defined.

For the third subsystem group, we created the Email services using Gmail as SMTP client so our testing focused in check that Emails generated were received by the correct recipients with the defined template and attachments if the message was generated with them.

***Frontend UI (Client Subsystem) and backend Web Services***

**Sunny** –

*PreCond:* admin has judge/student role and clicks the right button.

*Result:* Admin presented with judge or student screen. -- PASS

**Rainy** –

*PreCond:* admin does not have more roles.

*Result:* Admin does not see more role buttons. – PASS

## 7.3. Evaluation of Tests

|  |  |  |
| --- | --- | --- |
| Test Case ID | Description | Result |
|  | Fetch Known Key | Passed |
| M011 | Fetch Unkown Key | Passed |
| M012 | ValidCMSUpdate | Passed |
| M013 | InValidCMSUpdate | Passed |
| M014 | SendValidEmail | Passed |
| M015 | DontSendInvalidEmail | Failed |

# 8. Glossary