

## Table of Contents

<b>CHAPTER 1 PROPOSAL .....</b>	15
<b>1. SCOPE.....</b>	15
<b>1.1 Background.....</b>	15
<b>1.2 Solution.....</b>	16
<b>1.3 About us.....</b>	16
<b>1.4 Key personnel.....</b>	17
<b>1.5 Contact us.....</b>	17
<b>2. REFERENCE.....</b>	17
<b>3. PLANS.....</b>	18
<b>3.1 Work Breakdown Structure .....</b>	18
<b>3.2 Activity Schedule .....</b>	21
<b>3.3 Activity Graph.....</b>	24
<b>3.4 Development tasks.....</b>	30
<b>3.5 Project Schedule.....</b>	33
<b>3.6 Documentation Management Plan .....</b>	35
<b>3.7 Data Management Plan .....</b>	37
<b>3.8 Risk Management Plan .....</b>	37
<b>3.9 Test Plan .....</b>	38
<b>3.10 Security Plan.....</b>	39
<b>3.11 Deliverables.....</b>	39
<b>4. ESTIMATES.....</b>	40
<b>4.1 Estimate Size and Cost .....</b>	40
<b>4.2 Benefit .....</b>	42
<b>5. TERMS OF ACCEPTANCE.....</b>	42
<b>6. TERMS AND CONDITIONS .....</b>	42
<b>7. TERMS OF PAYMENT .....</b>	43
<b>8. WARRANTY .....</b>	44
<b>9. APPENDIX .....</b>	45
<b>CHAPTER 2. REQUIREMENT DOCUMENTATION .....</b>	51
<b>1. INTRODUCTION .....</b>	51
<b>1.1 Purpose.....</b>	51

<b>1.3 Definitions .....</b>	52
<b>1.4 System Overview.....</b>	54
<b>1.5 Reference.....</b>	55
<b>1.6 Overview of document.....</b>	55
<b>2. GENERAL DESCRIPTION .....</b>	55
<b>2.1 Product Perspective .....</b>	55
<b>2.2 Product Functions.....</b>	55
<b>2.3 User Characteristics.....</b>	56
<b>2.4 Design and Implementation Constraints.....</b>	57
<b>3. SOFTWARE REQUIREMENT .....</b>	59
<b>3.1 Functional Requirements .....</b>	59
<b>3.2 Non Functional Requirements .....</b>	91
<b>4. LOG OF MEETINGS.....</b>	93
<b>5. CHANGE OF CONTROL.....</b>	94
<b>6. APPENDIX .....</b>	95
<b>Appendix A. FWBS.....</b>	95
<b>Appendix B. Project Schedule .....</b>	98
<b>Appendix C. Terms of Acceptance .....</b>	102
<b>Appendix D. Terms and Condition .....</b>	102
<b>CHAPTER 3. DESIGN DOCUMENTATION .....</b>	108
<b>1. INTRODUCTION .....</b>	108
<b>1.1 Purpose.....</b>	108
<b>1.2 Major Problems.....</b>	108
<b>1.3 Project Goals.....</b>	108
<b>1.4 Definitions, Acronyms, and Abbreviations.....</b>	109
<b>1.5 Reference.....</b>	110
<b>1.6 Overview.....</b>	110
<b>2. DESIGN OVERVIEW.....</b>	111
<b>2.1 System Architecture.....</b>	111
<b>2.2 System Operation.....</b>	112
<b>3. DESIGN PRIORITY.....</b>	113
<b>3.1 Design priority table.....</b>	113
<b>3.2 Development and Execution Environment .....</b>	114

<b>3.3 Programming tools .....</b>	115
<b>3.4 Naming and Coding Standards.....</b>	116
<b>3.5 Exception Handling.....</b>	118
<b>3.6 Fault Tolerance.....</b>	119
<b>3.7 Design Constraints.....</b>	119
<b>4. DATA DESIGN.....</b>	121
<b>    1.1.1 Registration of patron.....</b>	122
<b>        1.1.1.1 waiver form .....</b>	125
<b>        1.1.1.2 user validation.....</b>	128
<b>        1.1.1.3 payment method.....</b>	129
<b>        1.1.1.4 Update user information .....</b>	131
<b>        1.1.2 Log In.....</b>	133
<b>        1.1.2.2 employee.....</b>	134
<b>            1.1.2.2.1.1 Register class .....</b>	137
<b>            1.1.2.2.1.2 EMS system .....</b>	138
<b>                1.1.2.2.1.2.1 Schedule class for certification.....</b>	140
<b>                1.1.2.2.1.2.2 Send Notifications.....</b>	141
<b>                1.1.2.2.1.3 certification management .....</b>	142
<b>                1.1.2.2.2.2 check in management .....</b>	144
<b>                1.1.2.2.2.5 request admin for user suspension.....</b>	145
<b>            1.1.2.3 admin.....</b>	147
<b>                1.1.2.3.1 staff management.....</b>	149
<b>                1.1.2.3.2 Admin for user suspension .....</b>	151
<b>                1.1.2.3.3 Reservation .....</b>	153
<b>                1.1.2.3.4 EMS system .....</b>	155
<b>                1.1.2.3.5 List Serv .....</b>	156
<b>                1.1.2.3.6 report generation.....</b>	158
<b>        1.2 Inventory .....</b>	159
<b>5. TOP LEVEL DESIGN.....</b>	162
<b>    5.1 Performance constraints.....</b>	162
<b>    5.2 Fault handling approach .....</b>	163
<b>6. MEDIUM LEVEL DESIGN .....</b>	164
<b>    1.1.1 Registration of patron.....</b>	164

<b>1.1.1.1 waiver form .....</b>	166
<b>1.1.1.2 user validation.....</b>	167
<b>1.1.1.2.d) Random ID Generation.....</b>	169
<b>1.1.1.3 payment method.....</b>	170
<b>1.1.1.4 Update user information .....</b>	172
<b>1.1.2 Log In.....</b>	172
<b>1.1.2.2 employee.....</b>	174
<b>1.1.2.2.1.1 Register class.....</b>	175
<b>1.1.2.2.1.2 EMS system .....</b>	178
<b>1.1.2.2.1.2.2 Send Notifications.....</b>	181
<b>1.1.2.2.1.3 certification management .....</b>	183
<b>1.1.2.2.2.2 check in management.....</b>	184
<b>1.1.2.2.2.5 request admin for user suspension.....</b>	186
<b>1.1.2.3 admin.....</b>	187
<b>1.1.2.3.1 staff management.....</b>	189
<b>1.1.2.3.2 admin for user suspension.....</b>	192
<b>1.1.2.3.3 Reservation .....</b>	193
<b>1.1.2.3.4 EMS system .....</b>	195
<b>1.1.2.3.5 List Serv .....</b>	199
<b>1.1.2.3.6 report generation.....</b>	201
<b>1.2 inventory .....</b>	203
<b>7. Appendix.....</b>	205
<b>Appendix A. Data dictionary .....</b>	205
<b>Appendix B. FWBS .....</b>	207
<b>Appendix C. Meeting Log .....</b>	211
<b>1. INTRODUCTION .....</b>	217
<b>1.1 Purpose.....</b>	217
<b>1.2 System Overview.....</b>	218
<b>1.3 Test Principle .....</b>	219
<b>1.4 Definitions, Acronyms, and Abbreviations.....</b>	219
<b>1.5 Reference.....</b>	220
<b>1.6 Overview of Document .....</b>	220
<b>2. HARDWARE AND SOFTWARE FOR TESTING .....</b>	221

<b>2.1 Hardware .....</b>	221
<b>2.2 Software .....</b>	221
<b>3. TEST SCHEDULE.....</b>	222
<b>4. TEST SETS .....</b>	223
<b>5. ERROR HANDLING POLICY.....</b>	225
<b>6. INDIVIDUAL TEST CASES.....</b>	226
<b>    1.1.1 Registration of patron.....</b>	226
<b>    1.1.2 Log In.....</b>	229
<b>    1.1.2.2.2 check in management.....</b>	230
<b>7. Log of Meetings .....</b>	232
<b>8. Project Acceptance.....</b>	232
<b>CHAPTER 5 SYSTEM TEST PLAN.....</b>	238
<b>    1. INTRODUCTION .....</b>	238
<b>        1.1 Purpose.....</b>	238
<b>        1.2 Scope of Product .....</b>	238
<b>        1.3 Definitions, Acronyms, and Abbreviations.....</b>	238
<b>        1.4 Reference.....</b>	239
<b>        1.5 Overview of Document .....</b>	239
<b>    2. TEST ITEMS .....</b>	239
<b>    3. SOFTWARE TESTING APPROACH .....</b>	240
<b>        3.1 Software testing team.....</b>	240
<b>        3.2 Software testing process.....</b>	241
<b>        3.3 Hardware and software requirements.....</b>	241
<b>            3.3.1 Hardware requirements.....</b>	241
<b>            3.3.2 Software requirements .....</b>	242
<b>    4. TRANSACTION FLOW TESTING .....</b>	243
<b>        4.1 Test1. Registration of patron .....</b>	243
<b>            4.1.1 Sprint &amp; FWBS .....</b>	243
<b>            4.1.2 Technique 1: Functionality and Procedure.....</b>	243
<b>            4.1.3 Technique 2: Causes and effect .....</b>	244
<b>            4.1.4 Technique 2: Negative Testing.....</b>	245
<b>            4.1.5 Boundary Value Analysis &amp; Equivalence Partitioning.....</b>	246
<b>            4.1.6 Merged Test Cases .....</b>	247

<b>4.1.7 Requirement Traceability Matrix .....</b>	250
<b>4.2 Test 2. Log In.....</b>	255
<b>4.2.1.Sprint &amp; FWBS .....</b>	255
<b>4.2.1.2 Functionality and Procedure.....</b>	255
<b>4.2.2 Technique 1: Positive Testing .....</b>	256
<b>4.2.3 Technique 2: Negative Testing.....</b>	256
<b>4.2.4 Technique 3: Decision Table and Cause-Effective Graph.....</b>	257
<b>4.2.5 Equivalence Partitioning and BVA .....</b>	259
<b>4.2.6 Merge Test Cases.....</b>	261
<b>4.2.7 Requirement Traceability Matrix .....</b>	262
<b>4.3 Test 3. Check-in management.....</b>	263
<b>4.3.1 Sprint &amp; FWBS .....</b>	263
<b>4.3.2 Functionality and Procedure.....</b>	263
<b>4.3.3 Technique 1: Boundary value analysis .....</b>	264
<b>4.3.4 Technique 2: Negative Testing.....</b>	265
<b>4.3.5 State based graph.....</b>	266
<b>4.3.6 Merge Test Cases.....</b>	266
<b>4.3.7 Requirement Traceability Matrix .....</b>	267
<b>4.4 Test 4 Waiver form.....</b>	267
<b>4.4.1 Sprint &amp; FWBS .....</b>	267
<b>4.4.2 Functionality and Procedure.....</b>	267
<b>4.4.3 Technique 2: Negative Testing.....</b>	268
<b>4.4.4 Decision Process .....</b>	268
<b>4.4.5 Requirement Traceability Matrix .....</b>	270
<b>4.5 Test 5. Class Registration .....</b>	270
<b>4.5.1 Sprint &amp; FWBS .....</b>	270
<b>4.5.2 Functionality and Procedure.....</b>	270
<b>4.5.3 Technique 1: Boundary Value Analysis &amp; Equivalence partitioning.....</b>	271
<b>4.5.4 Technique 2: Negative Testing.....</b>	273
<b>4.6 Test 6. Daily Note .....</b>	277
<b>4.6.1 Sprint &amp; FWBS .....</b>	277
<b>4.6.2 Functionality and Procedure.....</b>	277
<b>4.6.3 Technique 1: Negative Testing.....</b>	278

<b>4.6.4 State Based Graph .....</b>	278
<b>4.6.5 Merge Test Cases.....</b>	279
<b>4.6.6 Requirement Traceability Matrix .....</b>	280
<b>4.7 Test 7. Payment .....</b>	280
<b>4.7.1 Sprint &amp; FWBS .....</b>	280
<b>4.7.2 Functionality and Procedure.....</b>	280
<b>4.7.3 causes and effects .....</b>	281
<b>4.7.4 Technique 1: Negative Testing.....</b>	281
<b>4.7.5 Merge Test Cases.....</b>	281
<b>4.8 Test 8 Schedule class .....</b>	282
<b>4.8.1 Sprint &amp; FWBS .....</b>	282
<b>4.8.2 Functionality and Procedure.....</b>	282
<b>4.8.3 State Based Graph .....</b>	283
<b>4.8.4 Technique 1: Boundary Value analysis &amp; Equivalence partitioning .....</b>	283
<b>4.8.5 Technique 2: Negative Testing.....</b>	285
<b>4.8.6 Merge Test Cases.....</b>	285
<b>4.8.7 Requirement Traceability Matrix .....</b>	287
<b>4.9 Test 9 Send notification .....</b>	288
<b>4.9.1 Sprint &amp; FWBS .....</b>	288
<b>4.9.2 Functionality and Procedure.....</b>	288
<b>4.9.3 Technique 1: Negative Testing.....</b>	288
<b>4.9.4 Merge Test Cases.....</b>	289
<b>4.9.5 Requirement Traceability Matrix .....</b>	290
<b>4.10 Test 10 Certification management .....</b>	290
<b>4.10.1 Sprint &amp; FWBS .....</b>	290
<b>4.10.2 Functionality and Procedure.....</b>	291
<b>4.10.3 Technique 2: Negative Testing.....</b>	291
<b>4.10.4 Merge Test Cases.....</b>	291
<b>4.10.5 Requirement Traceability Matrix .....</b>	292
<b>4.11 Test 11 Request Admin for suspension .....</b>	292
<b>4.11.1 Sprint &amp; FWBS .....</b>	292
<b>4.11.2 Functionality and Procedure.....</b>	292
<b>4.11.3 State Based Graph.....</b>	293

<b>4.11.4 Technique 1: Boundary Value Analysis &amp; Equivalence partitioning .....</b>	293
<b>4.11.5 Technique 2: Negative Testing.....</b>	294
<b>4.11.6 Merge Test Cases.....</b>	295
<b>4.12 Test 12 Suspend patron.....</b>	297
<b>4.12.1 Sprint &amp; FWBS .....</b>	297
<b>4.12.2 Functionality and Procedure.....</b>	297
<b>4.12.3 State Based Graph.....</b>	298
<b>4.12.4 Negative Testing.....</b>	298
<b>4.12.5 Merge Test Cases.....</b>	299
<b>4.12.6 Requirement Traceability Matrix .....</b>	300
<b>4.13 Test 13. Report generation .....</b>	300
<b>4.13.1 Sprint &amp; FWBS .....</b>	300
<b>4.13.2 Functionality and Procedure.....</b>	300
<b>4.13.3 Technique 1: Positive Testing .....</b>	300
<b>4.13.4 Technique 2: Negative Testing.....</b>	301
<b>4.13.5 Merge Test Cases.....</b>	302
<b>4.13. 6 Requirement Traceability Matrix.....</b>	303
<b>4.14 Test 14. Inventory .....</b>	303
<b>4.14.1 Sprint &amp; FWBS .....</b>	303
<b>4.14.2 Functionality and Procedure.....</b>	303
<b>4.14.3 Technique 2 Negative Testing.....</b>	304
<b>4.14.4 Merge Test Cases.....</b>	304
<b>5. NON-FUNCTIONAL REQUIREMENT TESTING .....</b>	305
<b>5.1 Compatibility test.....</b>	305
<b>5.2 Regression test with policy .....</b>	306
<b>5.3 Backup and Recovery test.....</b>	307
<b>5.4 Stress and performance test.....</b>	307
<b>6. UNIT TESTING .....</b>	308
<b>6.1 Test 1 log in.....</b>	308
<b>6.1.1 Statement Coverage.....</b>	309
<b>6.1.2 Branch Coverage .....</b>	310
<b>6.1.3 McCabe's .....</b>	311
<b>6.1.4 Principal path .....</b>	311

<b>6.2 Test 2 Registration validation .....</b>	315
<b>6.2.1 Statement Coverage.....</b>	316
<b>6.2.2 Branch Coverage .....</b>	323
<b>6.2.3 McCabe's .....</b>	325
<b>6.2.4 Principal path .....</b>	325
<b>6.3 Test 3 Check-in validation .....</b>	334
<b>6.3.1 Statement Coverage.....</b>	334
<b>6.3.2 Branch Coverage .....</b>	334
<b>6.3.3 McCabe's .....</b>	336
<b>6.3.4 Principal path .....</b>	336
<b>7. INTEGRATION .....</b>	340
<b>7.1 McCabe's Complexity Graph.....</b>	341
<b>7.2 Principal path.....</b>	341
<b>7.3 Compound condition.....</b>	345
<b>7.4 Domain test .....</b>	349

## **Letter of proposal**

**1400 8th St.**

**Nester Center**

**Brookings, SD 57006**

**Subject: Proposal for e-Climbing system**

**09/15/2016**

**Dear Mr. Parks**

**Thank you for choosing Nester Software. We are very grateful and delighted to have this opportunity to respond your requirement for Wall Climbing Management System that helps to expand your business and increase your department benefits effectively.**

**Attached is the detailed proposal including the major focus on the Software Development Lifecycle, Deliverables, Project Management Process and Project Delivery Details. The software will cost \$200,000 to design, develop and test the software based on our estimate model. All of deliverables will be delivered before 12/1/2016.**

**We look forward to discussing the proposal along with the finalization of the modules. The proposal will be valid from September 15, 2016 for a period of 30 days once you approve it. Please feel free to contact us if you have any questions.**

**Thank you again for your consideration,**

**Yours Sincerely,**



**President of Nester Software Inc.**

# Proposal

## e-Climbing System

Prepared for:  
The Wall Climbing Center  
of  
South Dakota State University



Version 2.0  
Prepared by:  
**Nester Software. Inc**

1400 8th St. Nester Center  
Brookings, SD 57006  
09/2016

Approved for public release; distribution is unlimited

# **SOFTWARE DEVELOPMENT PROPOSAL**

**FOR**

## **E-CLIMBING SYSTEM**

**09/2016**

**Prepared By:**

**Nester Software Inc.**

**1800 8th St. Nester Center**

**Brookings, SD 57006**

**09/2016**

### **Team Information**

*Appala Chekuri*

---

**Software Project Manager**

**Signature**

*Hussein Otudi*

---

**Systems Engineer Manager:**

**Signature**

*Shaohu Zhang*

---

**Chief Executive Officer**

**Signature**

## Authoring & Approval

Name	Position & Title	Responsibility	Date
Shaohu Zhang	CEO Nester Software Inc.	Author	9/13/2016
Appala Chekuri	Software project manager Nester Software Inc.	Author	9/13/2016
Hussein Otudi	Systems Engineer Manager Nester Software Inc.	Author	9/13/2016
Justin Park	South Dakota State University	Approval	9/13/2016

## Reviewer

Name	Position & Title	Responsibility	Date
Shaohu Zhang	CEO Nester Software Inc.	Reviewer	9/13/2016
Hussein Otudi	Systems Engineer Manager Nester Software Inc.	Reviewer	9/13/2016

## Revision History

Date	Description	Revision	Editor
8/25/2016	Created the document	0	Hussein, Shaohu
8/25/2016	Added scope	0.1	Shaohu
8/26/2016	Added plan	0.2	Shaohu
8/26/2016	Update plan	0.3	Hussein, Appala
8/28/2016	Update plan	0.4	Shaohu
8/28/2016	Update plan	0.5	Shaohu
9/01//2016	Added estimate	0.6	Hussein, Appala Shaohu
9/01//2016	Update estimate	0.7	Appala, Hussein
9/01//2016	Added terms of acceptance	0.8	Hussein, Appala Shaohu

9/01/2016	Added terms and condition	0.9	Shaohu
9/13//2016	Review	1.0	Shaohu
9/13//2016	Review	2.0	Shaohu

**This deliverable has two hard copies. One is for Dr. Shin from South Dakota State University, and another one is submitted to Mr. Park for approval. Distribution is unlimited for public release.**

# CHAPTER 1 PROPOSAL

## 1. SCOPE

### 1.1 Background

The Wall Climbing Center currently uses a EMS system to maintain the schedule and reservation. This system is limited to the function for making schedule and reservation only by administrator. It has difficulty to keep track the customer history and equipment status. In addition, it is very labor-intensive produce monthly report.

Generally, e-Climbing system consists of the four modules:

- Intelligent check-in system
- Intelligent Reservation System
- Intelligent Inventory System
- Point of Sale System

The intelligent check-in system can do the verification for the user information such like suspension, certification and waiver. This module can monitor dates/times effectively. System will deny access until that suspension date is over.

The second module is the intelligent reservation system. our system incorporates a live electronic reservation system. Customers can view for available specific days and times and submit a reservation online. Our system also can keep track of customers and gather the information such as how often they come and when they come.

The intelligent inventory system is designed for manage all types of equipment and customer service. In our system, the function for equipment profile, levels of customer and search by category and by date/time is provided. With every item the administrator can change the price per item. The program automatically recalculates the entire items and gives an up-to-date cost and profit on each item. This e-Climbing software will make the operation easy.

The Point of Sales System (POS) provides dozens of features such as fast order, detailed report and user-friendly interface. It is convenient for manager to gather specified report information such as daily or monthly report of sales and visitor information. The payment functionality could support different credit card.

## **1.2 Solution**

Our solution is going to effectively and efficiently manage all aspects of the Climbing Wall at SDSU. E-Climbing System will provide an innovative solution to patron, employee and admin. This entirely Web-based application provides the following features.

The proposed system is easy to use. The users will be comfortable while using the system for sign-up and reservation. The admin can track visitor information, class schedule, POS sales and much more. E-climbing is available anywhere with an internet connection.

The system will be provided with full documentation that would ensure the access and troubleshooting. We will use our innovative solution to deliver a fast response e-Climbing system having compatibility with a big database.

## **1.3 About us**

Nester Software is a privately held company that was founded on September, 2000 by Shaohu Zhang, Appala Chekuri and Hussein Otudi. With more than 5300 global customers and cooperative cooperation, Nester offers a comprehensive and fully integrated platform services and engineered systems. Nester Software is an ISO 9001:2000 certified company with a highly competent and proficient workforce.

Nester Software has developed and provided world's best software and web applications to many companies across the worldwide. The applications include intelligent reservation systems, Online Sales System and Operation Support Systems. Its profit has increased tremendously owing to the great market demand of our products and services. In 2015, Nester worldwide annual revenue totaled \$20 billion.

Nester Software's easy-to-use software along with expert, responsive customer support has a daily impact on its customers' businesses. We make sure that our employees are trained well on the latest tools, trends and technologies in the market industry. We frequently keep ourselves updated and have the latest machines and tools so that our product is latest and cost effective. Nester Software helps its customers significantly reduce application development and operating costs.

With a decade of successful software development experience, we believe the creation of a user friendly, robust and functionally rich software solution is what we deliver the best. We are

confident that we will be able to create an exceptional system that will meet and exceed client's expectations and vision for the proposed software application.

#### **1.4 Key personnel**

Shaohu Zhang

CEO

E-mail: [shaohu.zhang@sdstate.edu](mailto:shaohu.zhang@sdstate.edu)

Phone: (605)592-0499

Appala Chekuri

Program manager

E-mail: [appala.chekuri@sdstate.edu](mailto:appala.chekuri@sdstate.edu)

Phone:(605)592-0991

Hussein Otudi

Software Analyzer

E-mail: [hussain.otudi@sdstate.edu](mailto:hussain.otudi@sdstate.edu)

Phone:(605)592-0069

#### **1.5 Contact us**

1400 8th St.

Nester Center

Brookings SD 57006

Phone: (800)666-8888

Fax: (800)666-8888

## **2. REFERENCE**

### **1. Software development plan**

<http://www.acqnotes.com/Attachments/Software%20Development%20Plan%20Template%20-%20SPAWAR.pdf>

### **2. Terms and Conditions for Microsoft Office 365**

<http://totterdells.ie/assets/files/Pdf-downloads/Terms-and-Conditions-for-Microsoft-Office-365.pdf>

### **3. COCOMO**

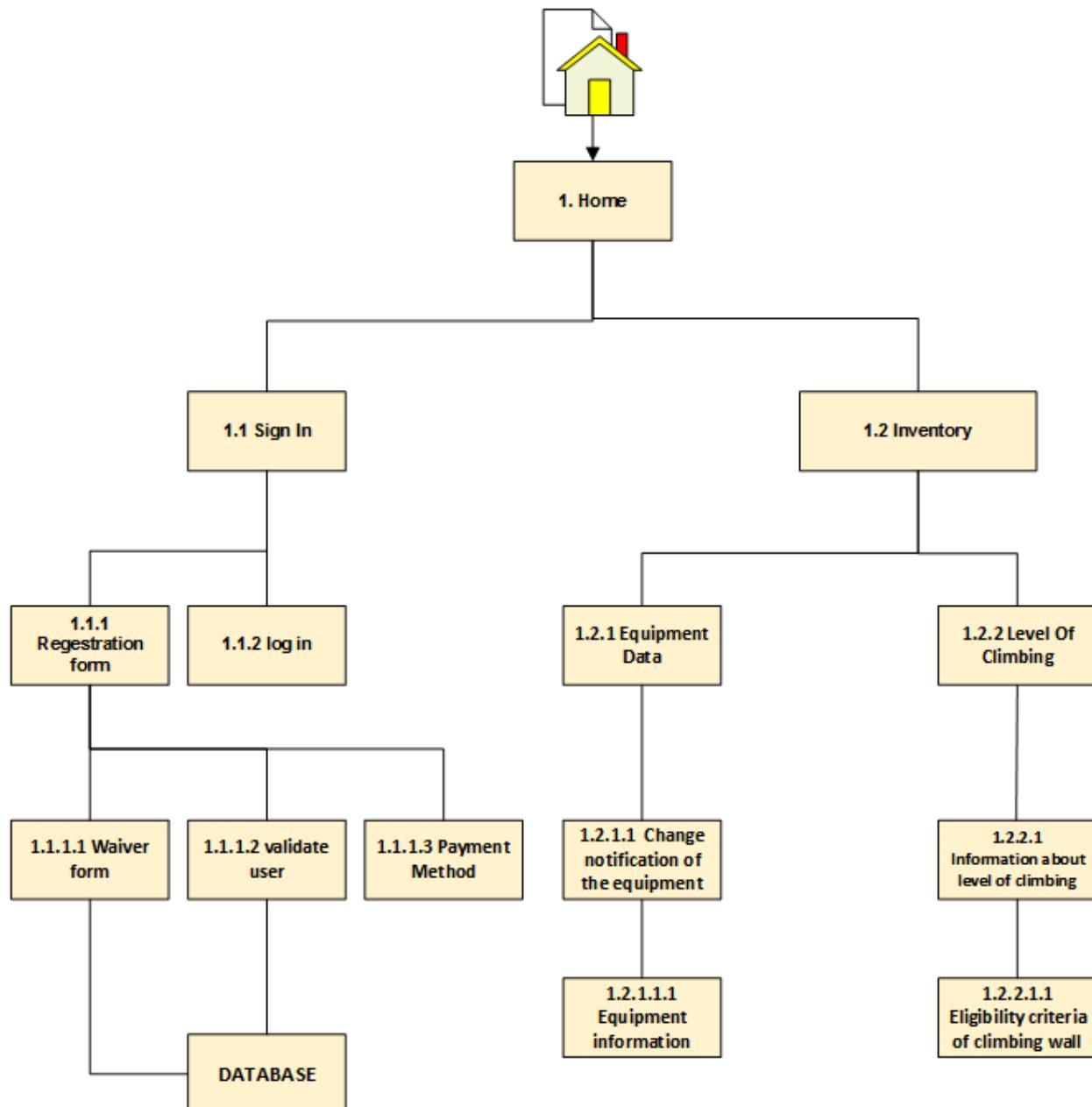
<https://en.wikipedia.org/wiki/COCOMO>

### **4. Guidelines for Effective Data Management Plans**

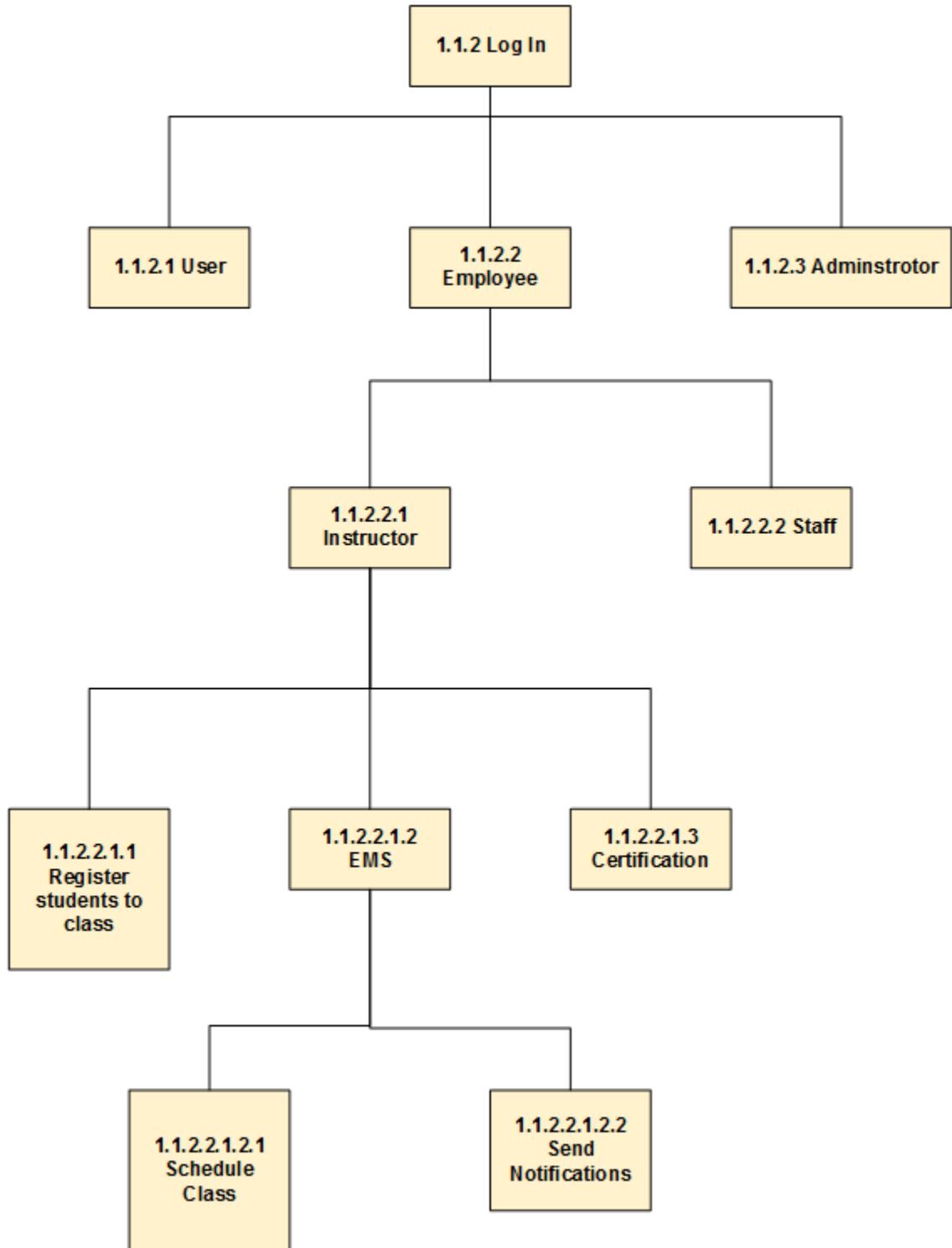
### 3. PLANS

#### 3.1 Work Breakdown Structure

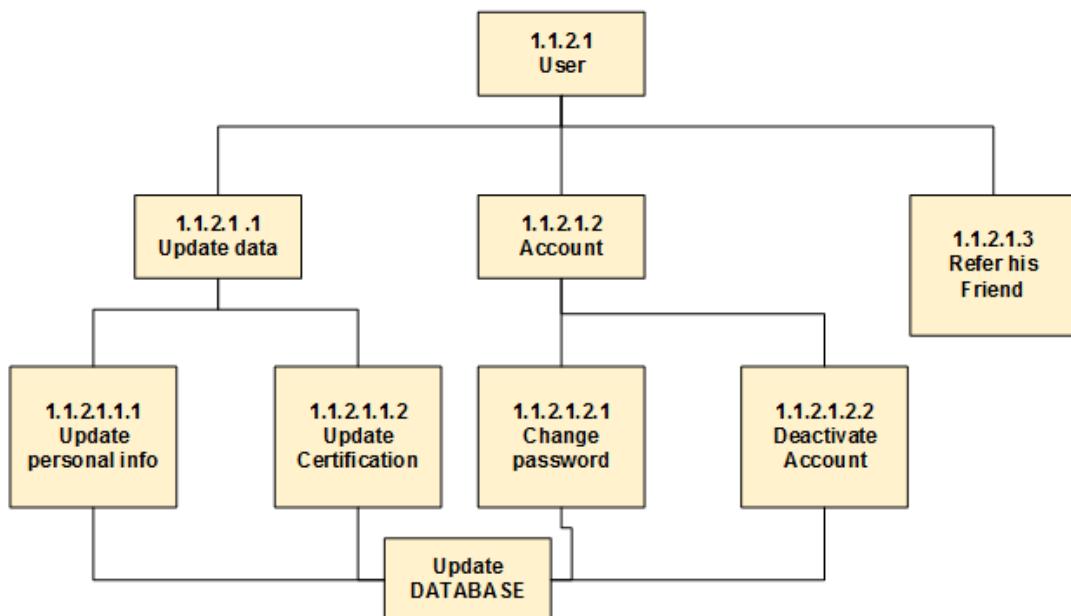
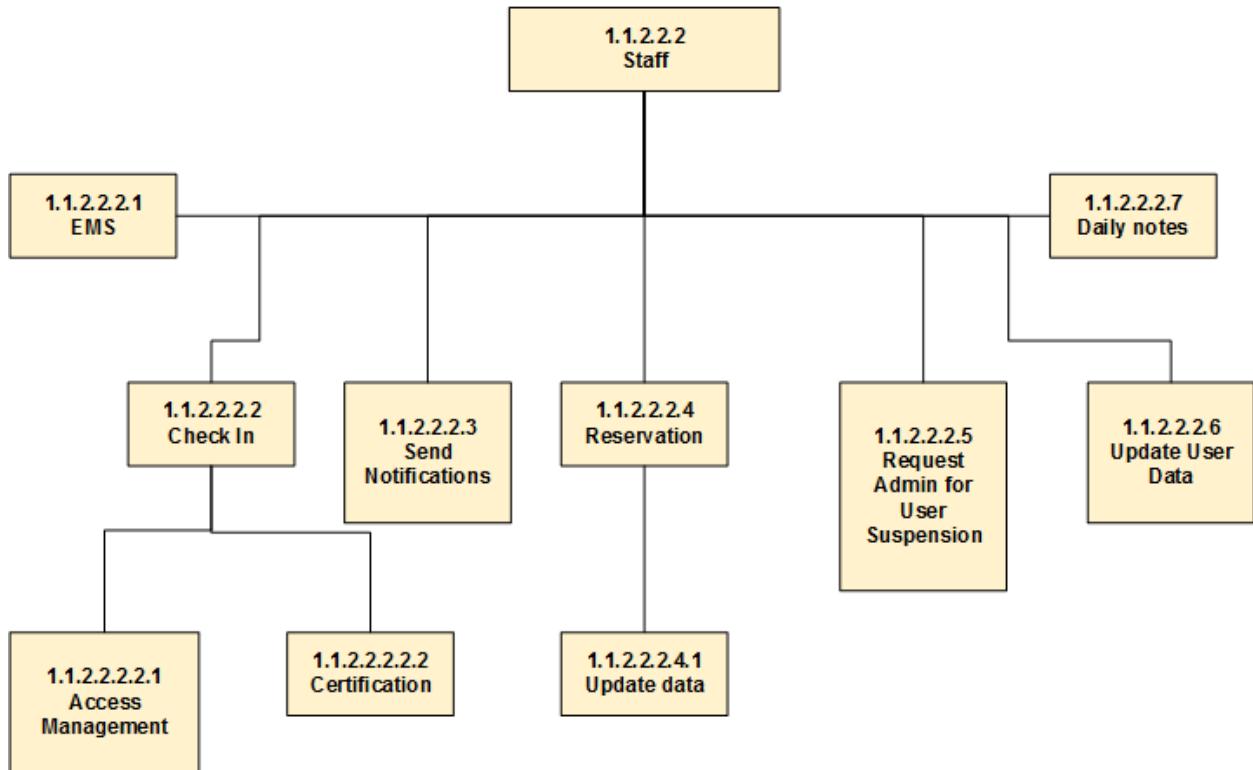
- Main Page:



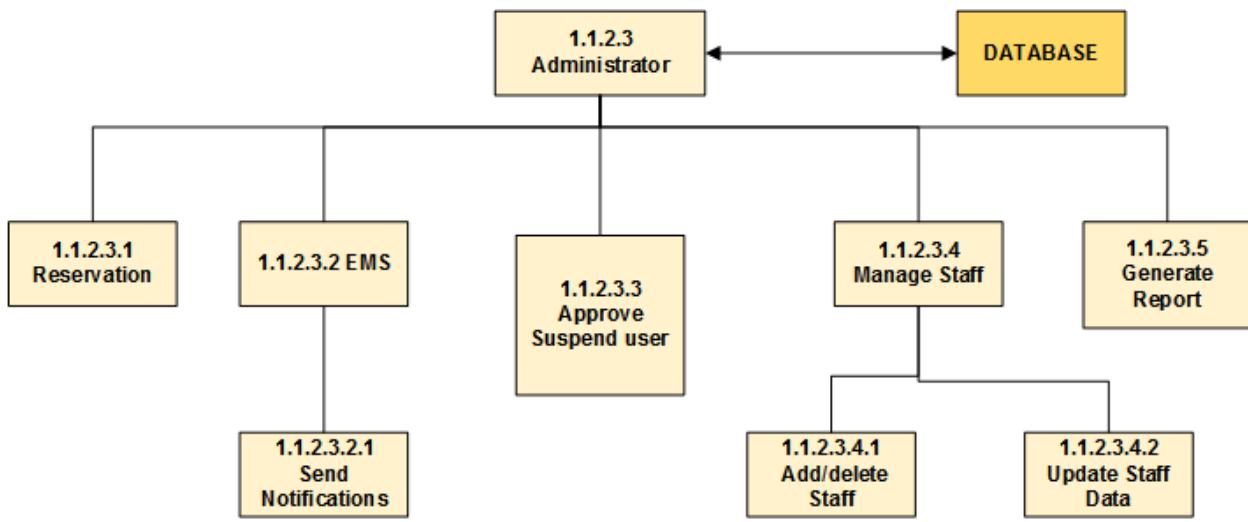
- Login Page:



- Staff and User



- Admin:



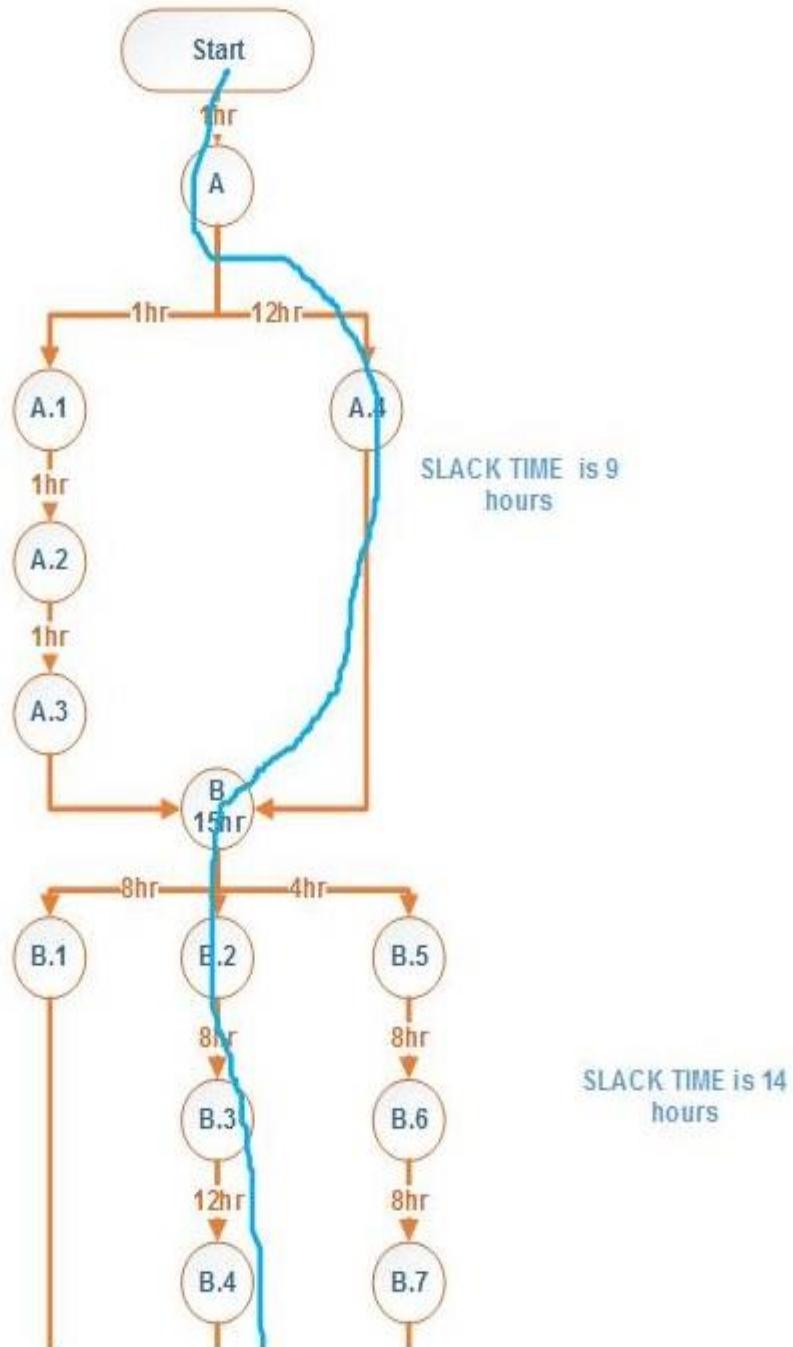
### 3.2 Activity Schedule

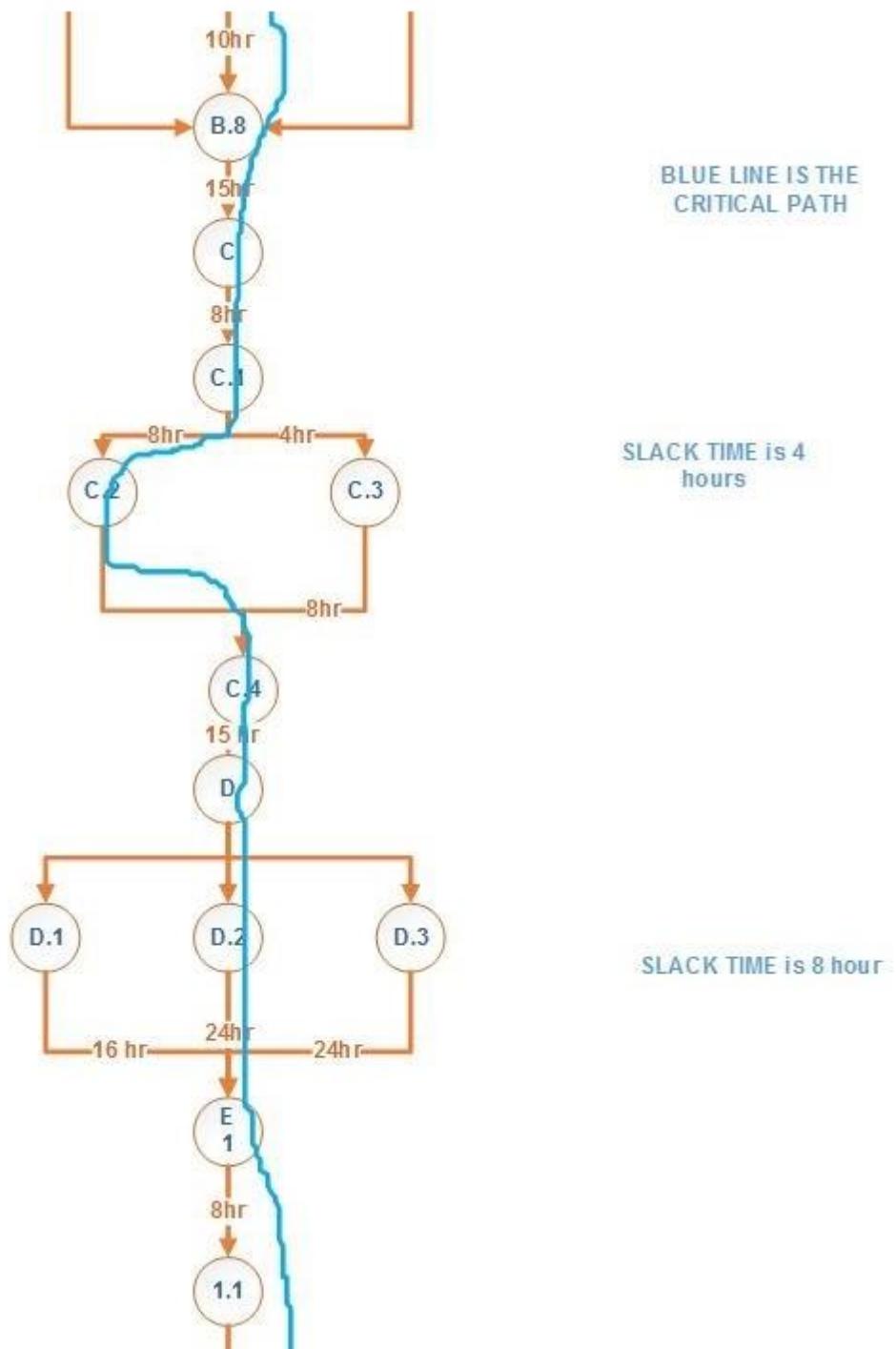
Number	Activity	Start	Finish	Duration	Slack Time	Critical
A	Market research and requirements analysis	8/25/2016	9/1/2016	1 hr	0 hr	Yes
A.1	Group Meeting --Start up	8/25/2016	8/25/2016	1 hr	11 hrs	No
A.2	Group Meeting --Company name --Logo	8/27/2016	8/27/2016	1 hr	10 hrs	NO
A.3	Interview customer	8/28/2016	8/28/2016	1 hr	9hrs	NO
A.4	Market research	8/25/2016	9/1/2016	12 hrs	0 hr	Yes
B	Proposal	9/1/2016	9/15/2016	15 hrs	0 hr	Yes
B.1	FWBS	9/1/2016	9/1/2016	8 hrs	8 hrs	NO
B.2	CPA	9/6/2016	9/6/2016	8 hrs	0 hr	Yes

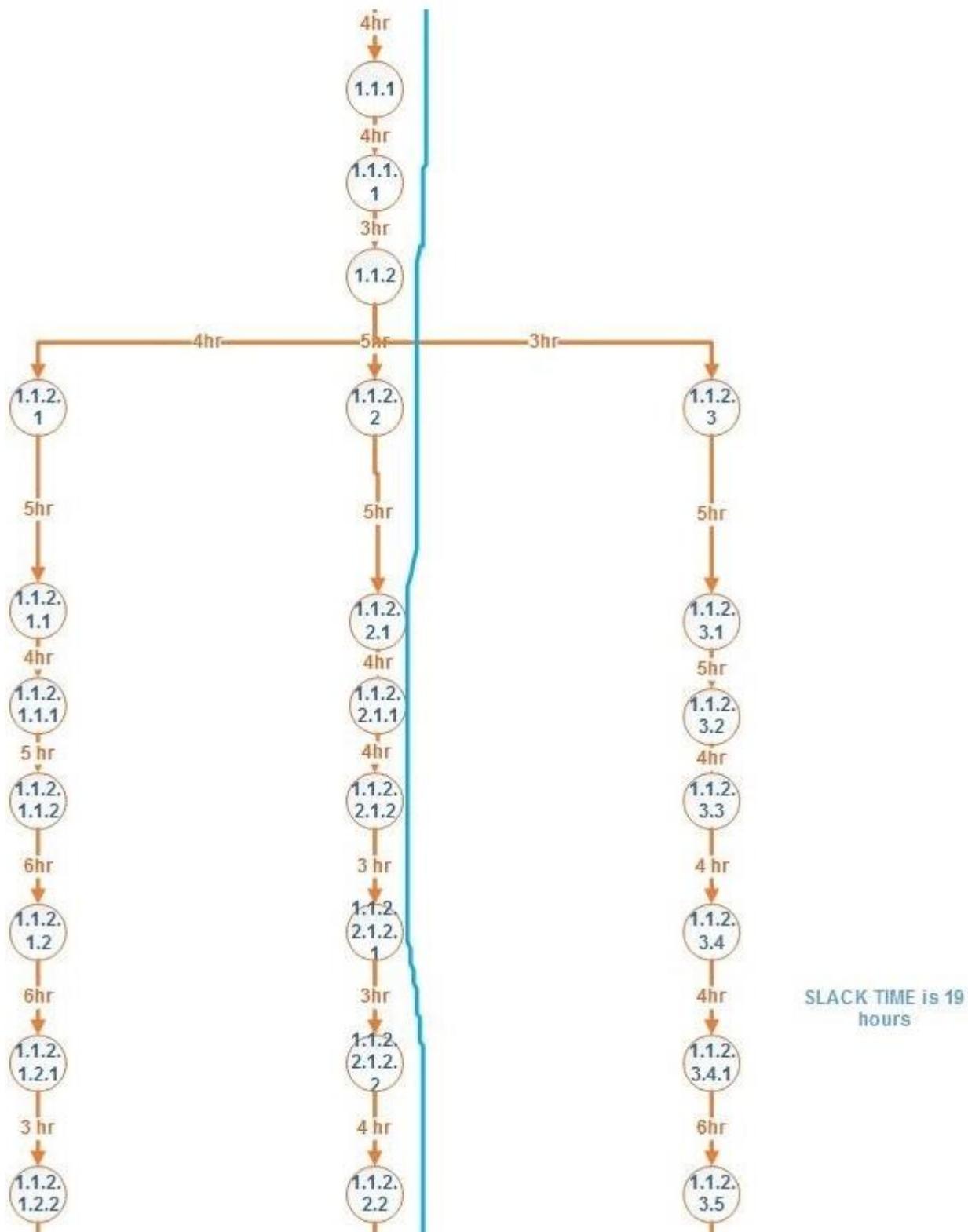
B.3	Activity Gantt chart	9/7/2016	9/8/2016	12 hrs	0 hr	Yes
B.4	Secure plan	9/6/2016	9/9/2016	10 hrs	0 hr	Yes
B.5	Scope complete	9/7/2016	9/9/2016	4 hrs	4 hrs	NO
B.6	Needs analysis	9/12/2016	9/13/2016	8 hrs	4 hrs	NO
B.7	Estimates	9/13/2016	9/13/2016	8 hrs	2 hrs	No
B.8	Proposal complete	9/14/2016	9/14/2016	8hrs	0 hr	Yes
C	Requirement document	9/15/2016	9/22/2016	15 hrs	0 hr	Yes
C.1	Group meeting --RD introduction --Split work	9/15/2016	9/15/2016	3 hrs	0 hr	Yes
C.2	Functional analysis	9/20/2016	9/21/2016	8 hrs	0 hr	Yes
C.3	Non-functional analysis	9/21/2016	9/21/2016	4 hrs	4 hrs	NO
C.4	Polish RD	9/22/2016	9/22/2016	8hrs	0 hr	Yes
D	Design document	9/23/2016	10/6/2016	15 hrs	0 hr	Yes
D.1	Review preliminary specifications	9/23/2016	9/26/2016	16 hrs	8 hrs	NO
D.2	Develop functional specifications	9/27/2016	10/6/2016	24 hrs	0 hr	Yes
D.3	Develop prototype	10/4/2016	10/6/2016	20 hrs	4 hrs	NO
E	Home interface design	10/14/2016	10/14/2016	8 hrs	0 hr	Yes
E.1	Admin interface design	10/15/2016	10/18/2016	24 hrs	0 hr	Yes
E.2	Employee interface design	10/18/2016	10/21/2016	24 hrs	0 hr	Yes
E.3	Patron interface design	10/21/2016	10/24/2016	24 hrs	0 hr	Yes
E.4	Check in system design	10/24/2016	10/25/2016	16 hrs	0 hr	Yes

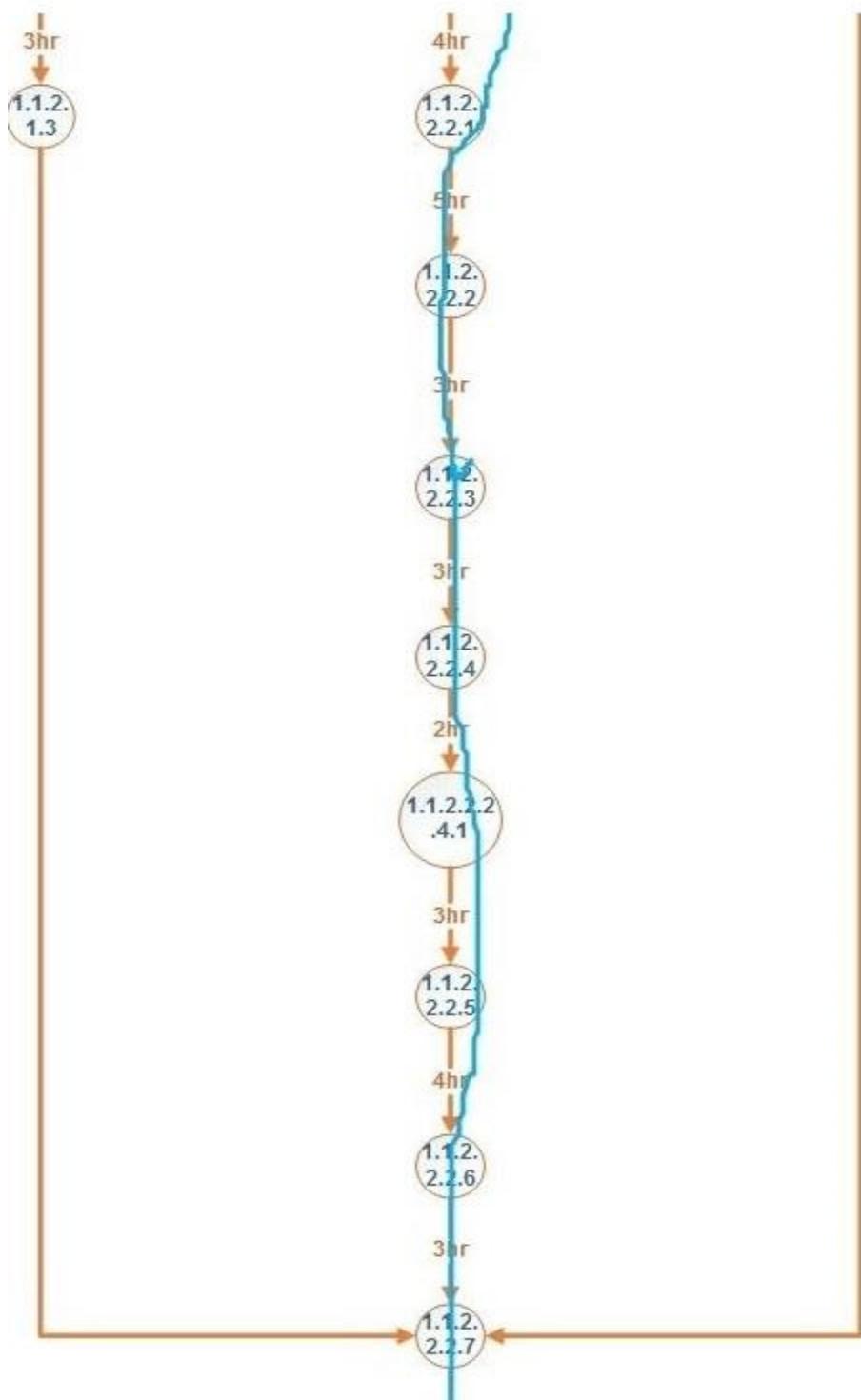
E.5	Reservation system design	10/25/2016	10/26/2016	16 hrs	0 hr	Yes
E.6	Inventory system design	10/26/2016	10/27/2016	16 hrs	0 hr	Yes
F	Level verification	10/28/2016	10/28/2016	8 hrs	0 hr	Yes
G	System test plan	10/7/2016	11/10/2016	16 hrs	0 hr	Yes
H	System integration	11/11/2016	11/15/2016	20 hrs	0 hr	Yes
I	Acceptance test plan	11/16/2016	11/17/2016	20 hrs	0 hr	Yes
J	User's guide	11/18/2016	11/20/2016	20 hrs	0 hr	Yes
K	Deliver the final project Source code	11/21/2016	12/1/2016	30 hrs	0 hr	Yes
L	Client Training	12/2/2016	12/2/2016	8 hrs	0 hr	Yes

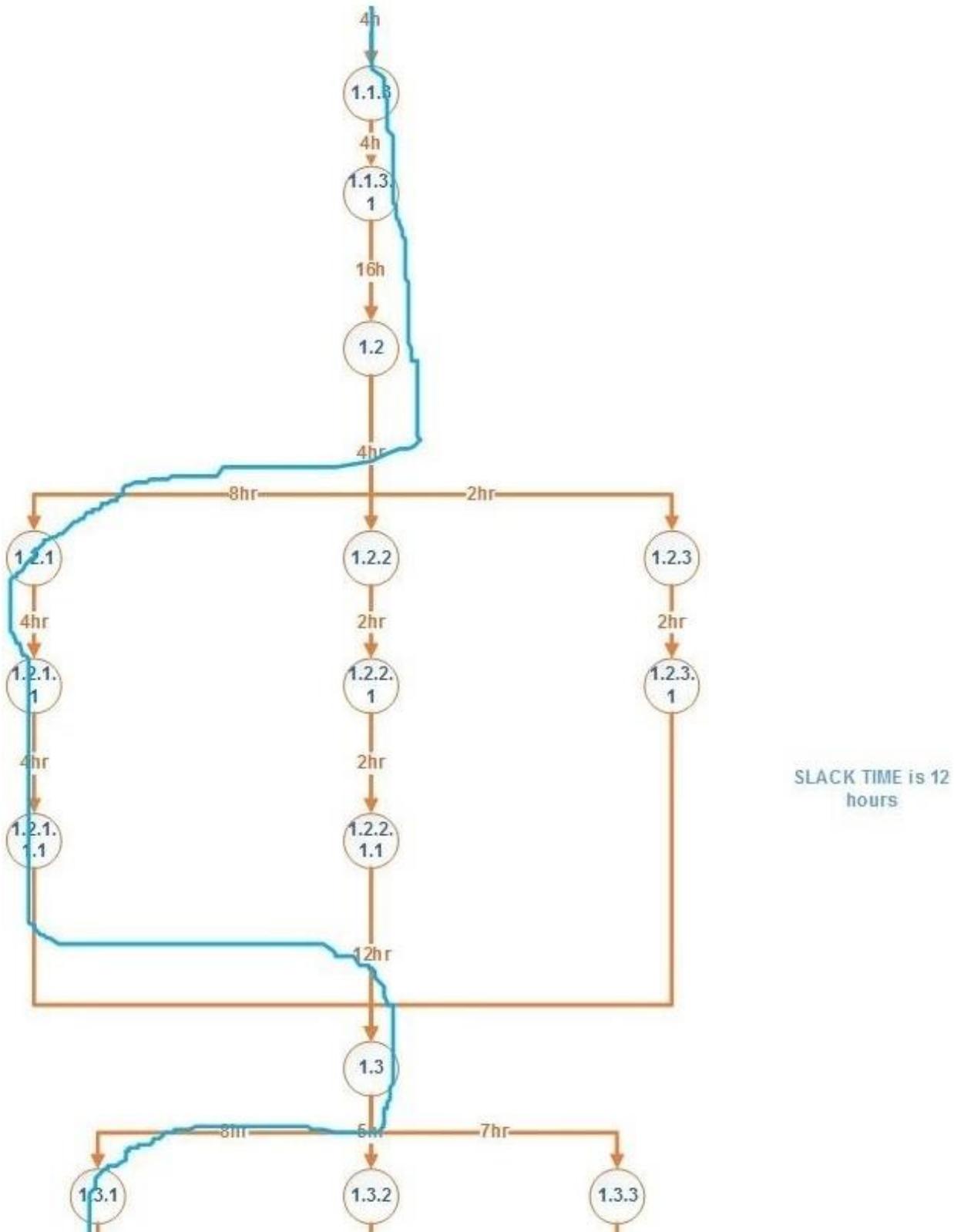
### 3.3 Activity Graph

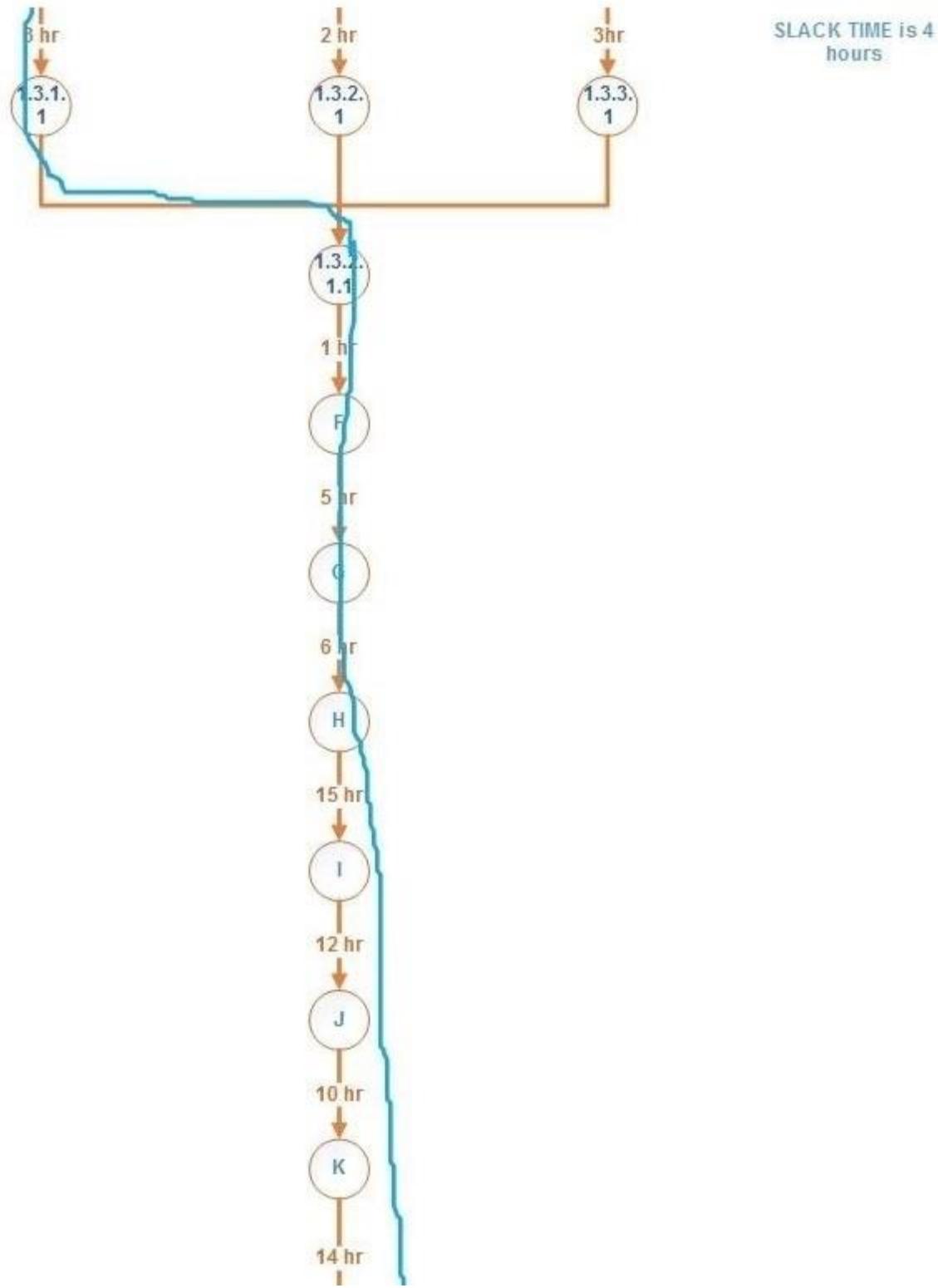


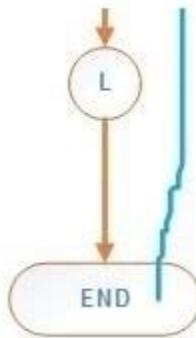












### 3.4 Development tasks

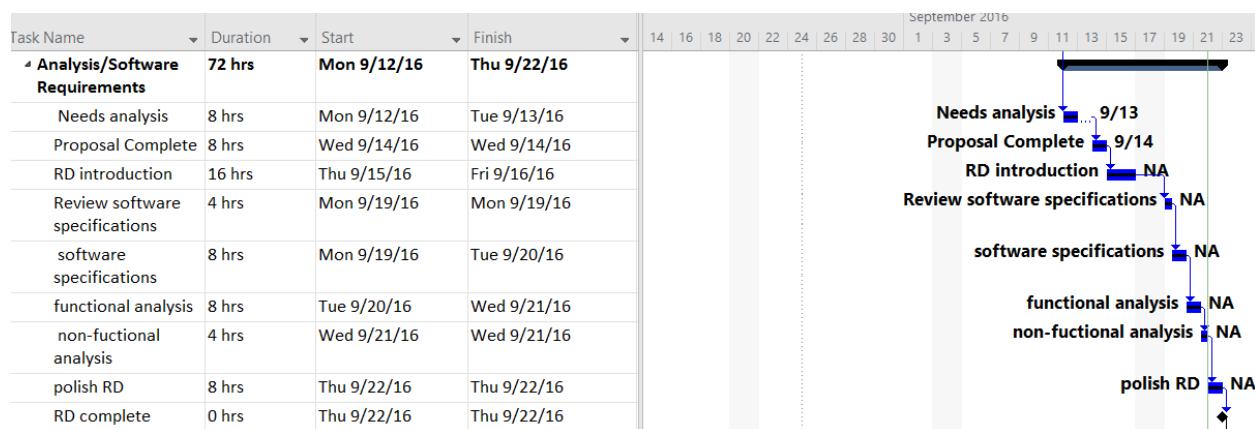
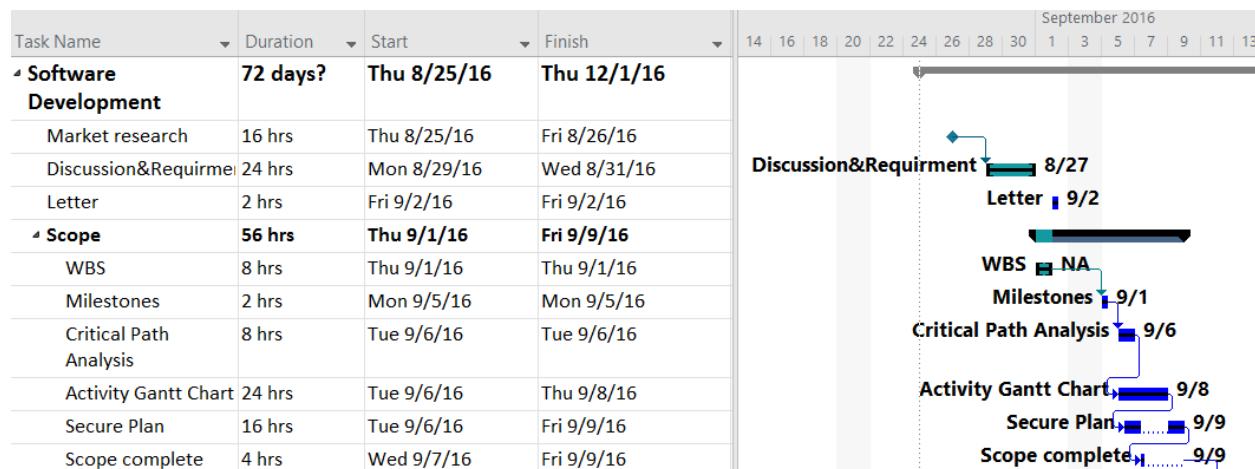
FWBS	TASK NAME	DURATION	SLACK	START	FINISH	Critical
1	Has the Main page what user will get on the screen	8 hrs	0 hr	10/14/16	10/15/16	Yes
1.1	This contains the login page	4 hrs	0 hr	10/15/16	10/15/16	Yes
1.1.1	The registration form	4 hrs	0 hr	10/17/16	10/17/16	Yes
1.1.1.1	Waiver form	4 hrs	0 hr	10/18/16	10/18/16	Yes
1.1.2	Log in of the User, Employee, Admin	3 hrs	0 hr	10/18/16	10/18/16	Yes
1.1.1.2	Validation of the user	4 hrs	4 hrs	10/19/16	10/19/16	No
1.1.3	Support desk and the information	4 hrs	4 hrs	10/19/16	10/20/16	NO
1.1.1.3	Payment method	4 hrs	4 hrs	10/19/16	10/19/16	NO
1.2	Inventory Control	16 hrs	0 hr	10/25/16	10/26/16	Yes
1.2.1	Equipment Data	8 hrs	0 hr	10/25/16	10/25/16	Yes
1.2.1.1	Change notification of the equipment	4 hrs	0 hr	10/25/16	10/25/16	Yes
1.2.1.1.1	Equipment information	4 hrs	0 hr	10/25/16	10/25/16	Yes

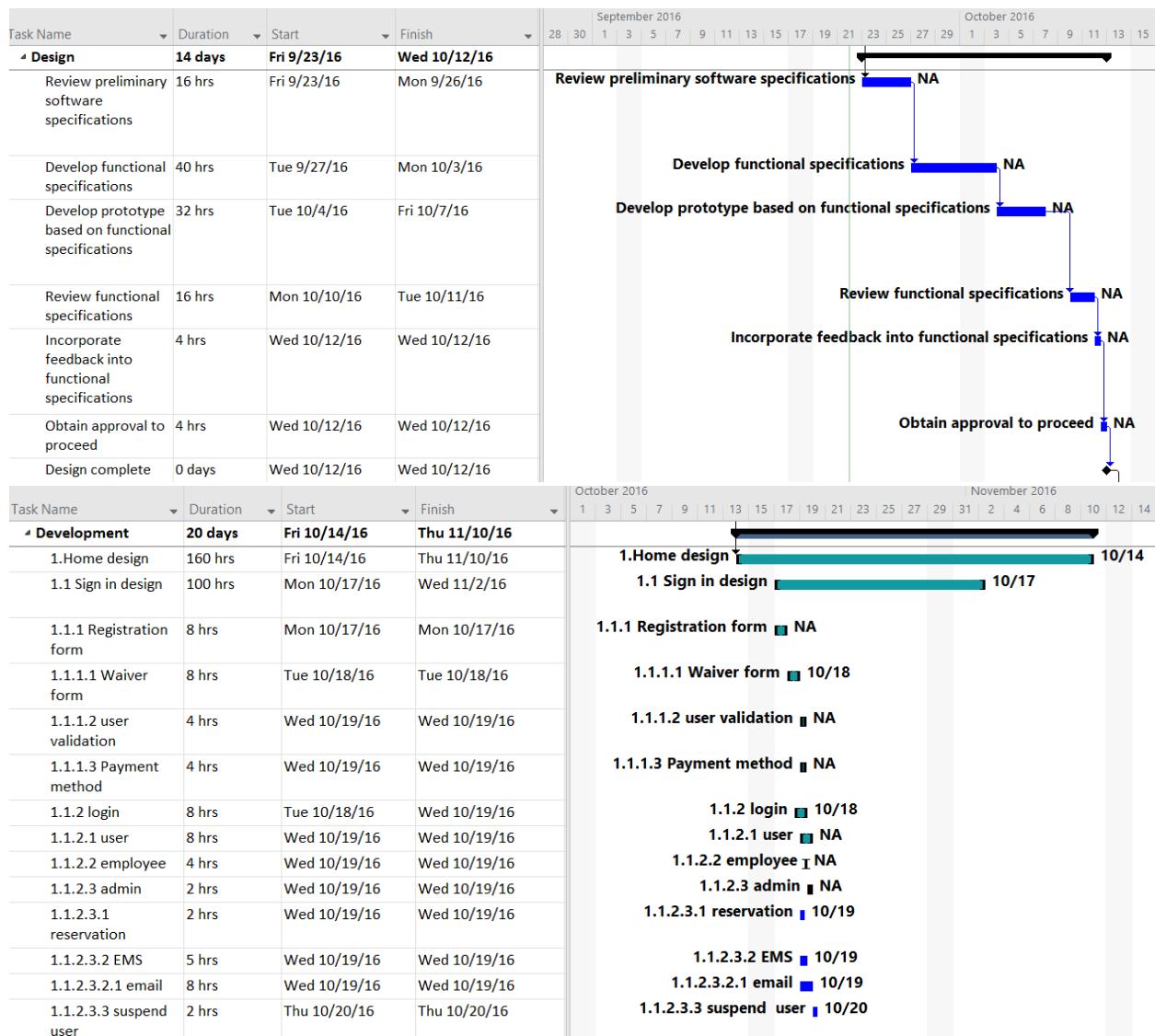
1.2.2	Level of climbing	4 hrs	4 hrs	10/28/16	10/28/16	NO
1.2.2.1	Information about level of climbing	2 hrs	2 hrs	10/28/16	10/28/16	NO
1.2.2.1.1	Eligibility criteria of climbing wall	2 hrs	4 hrs	10/28/16	10/28/16	NO
1.2.3	Internet search engine	2 hrs	6 hrs	10/31/16	10/31/16	NO
1.2.3.1	Search engine	2 hrs	6 hrs	10/31/16	10/31/16	NO
1.3	Point of Sale	5 hrs	0 hr	10/27/16	10/27/16	Yes
1.3.1	Equipment	8 hrs	0 hr	10/27/16	10/27/16	Yes
1.3.1.1	Date and price of equipment	3 hrs	0 hr	10/28/16	10/28/16	YES
1.3.2	Sell Equipment information	5 hrs	3 hrs	10/28/16	10/28/16	NO
1.3.2.1	Post equipment on internet	2 hrs	4 hrs	10/28/16	10/28/16	NO
1.3.2.1.1	Search engine	1 hr	1 hrs	10/29/16	10/29/16	NO
1.3.3	Budget Management	7 hr	1 hr	11/1/16	11/1/16	NO
1.3.3.1	Amount after selling equipment	3 hrs	4 hrs	11/1/16	11/1/16	NO
1.3.3.2	Amount of profit and loss on equipment	3 hrs	4 hrs	11/1/16	11/1/16	NO
1.1.2.3	Administrator	3 days	4 hrs	10/19/16	10/19/16	NO
1.1.2.3.1	Reservation	2 hrs	2 hrs	10/19/16	10/19/16	NO
1.1.2.3.2	EMS send notifications	5 hrs	3 hrs	10/19/16	10/19/16	NO
1.1.2.3.3	Approve suspend user	2 hrs	6 hrs	10/20/16	10/20/16	NO
1.1.2.3.4	Manage Staff	6 hrs	2 hrs	10/21/16	10/21/16	NO
1.1.2.3.4.1	Add New Staff	3 hrs	5 hrs	10/21/16	10/21/16	NO
1.1.2.3.4.2	Update Staff Data	1 hr	1 hr	10/21/16	10/21/16	NO

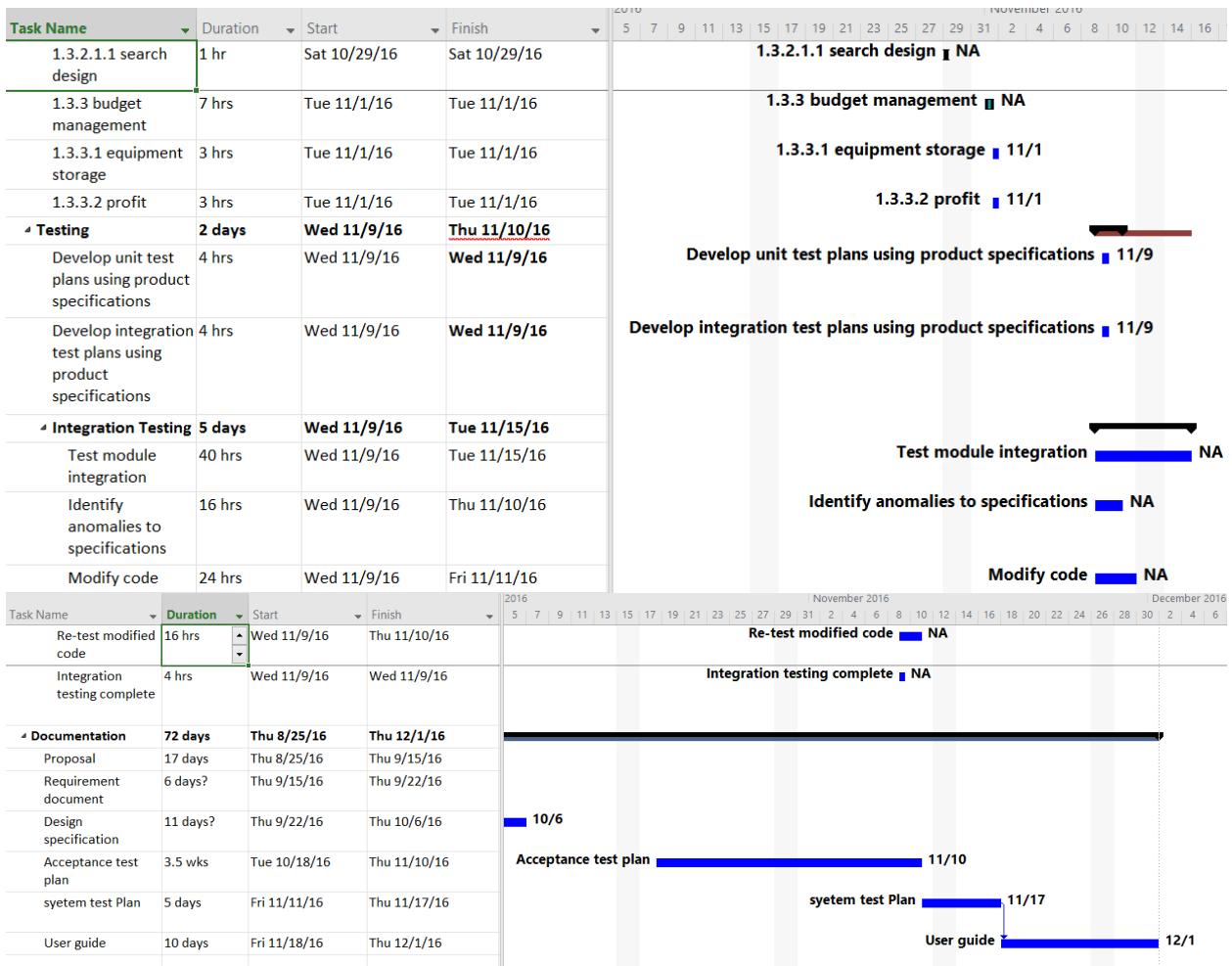
1.1.2.3.5	Generate Report	1 hr	1 hr	10/24/16	10/24/16	NO
1.1.2	Log In Requirement Page	3 hrs	0 hr	10/18/16	10/21/16	Yes
1.1.2.1	User login	3 hrs	5 hrs	10/19/16	10/19/16	NO
1.1.2.2	Employee login	5 hrs	0 hr	10/19/16	10/26/16	Yes
1.1.2.3	Administrator	3 hrs	7 hrs	10/19/16	10/26/16	NO
1.1.2.2.1	Instructor	3 hrs	0 hr	10/19/16	10/21/16	Yes
1.1.2.2.2	Staff	4 hrs	0 hr	10/20/16	10/20/16	Yes
1.1.2.2.1.1	Register students to class	2 hrs	1 day	10/20/16	10/20/16	NO
1.1.2.2.1.2	EMS	1 day	1 day	10/20/16	10/20/16	NO
1.1.2.2.1.2.1	Schedule Class	1 day	1 day	10/21/16	10/21/16	NO
1.1.2.2.1.2.2	Send Notifications	3 hrs	1 day	10/21/16	10/21/16	NO
1.1.2.2.1.3	Certification	4 hrs	1 day	10/21/16	10/21/16	NO
1.1.2.2.2	Staff	4 hrs	4 days	10/20/16	10/25/16	NO
1.1.2.2.2.1	EMS Scheduling System	1 day	1 day	10/21/16	10/21/16	NO
1.1.2.2.2.2	Check In	4 hrs	1 day	10/24/16	10/24/16	NO
1.1.2.2.2.2.1	Access Management	4 hrs	1 day	10/24/16	10/24/16	NO
1.1.2.2.2.2.2	Certification	1 hr	1 day	10/24/16	10/24/16	NO
1.1.2.2.2.3	Send notifications	4 hrs	1 day	10/24/16	10/24/16	NO
1.1.2.2.2.4	Reservation	6 hrs	1 day	10/24/16	10/24/16	NO
1.1.2.2.2.4.1	Update data of staff and students	2 hrs	1 day	10/24/16	10/24/16	NO
1.1.2.2.2.5	Request admin for User suspension	8 hrs	1 day	10/24/16	10/24/16	NO
1.1.2.2.2.6	Update user data	3 hrs	1 day	10/24/16	10/24/16	NO
1.1.2.2.2.7	Daily notes	1 hr	1 day	10/24/16	10/24/16	NO
1.1.2.1	User	3 hrs	3 days	10/19/16	10/21/16	NO
1.1.2.1.1	Update data	2 hrs	2 days	10/20/16	10/21/16	NO

1.1.2.1.1.1	Update personal information	5 hrs	1 day	10/20/16	10/20/16	NO
1.1.2.1.1.2	Update certification	5 hrs	1 day	10/21/16	10/21/16	NO
1.1.2.1.2	Account Privacy	5 hrs	2 hr	10/20/16	10/20/16	NO
1.1.2.1.2.1	Change Password	2 hrs	1 day	10/20/16	10/20/16	NO
1.1.2.1.2.2	Deactivate Account	2 hrs	1 day	10/20/16	10/20/16	NO
1.1.2.1.3	Refer his friend	5 hrs	2 hrs	10/20/16	10/20/16	NO

### 3.5 Project Schedule







### 3.6 Documentation Management Plan

To satisfy the required technical information throughout the development life cycle and maintenance of the proposed software system, all documentation should be in timely manner and meet the below criteria:

- The descriptions of the software management, processes and products should be clear, precise and consistent across the whole project documentation.
- The traceability of product for each phase of the development life cycle should be well maintained.

Meantime, each team member as the rotating program manager is going to be responsible to one document, respectively to make sure that innovation solutions will be well documented. The following deliverables that describes the details for the system.

Documentation	Plan	Deliverable
---------------	------	-------------

Proposal	<p>Shaohu:</p> <ul style="list-style-type: none"> <li>a) Background and solution;</li> <li>b) Estimates</li> <li>c) Milestone</li> <li>d) Project schedule</li> <li>e) Review</li> </ul> <p>Appala</p> <ul style="list-style-type: none"> <li>f) WBS</li> <li>g) Activity graph</li> <li>h) CPA</li> </ul> <p>Hussein</p> <ul style="list-style-type: none"> <li>i) Data management plan</li> <li>j) Risk management plan</li> <li>k) security plan</li> </ul> <p>Shaohu</p> <ul style="list-style-type: none"> <li>l)log of meeting</li> <li>m) Acceptance</li> <li>n) Terms and conditions</li> <li>o) Terms and payments</li> </ul>	9/15/2016
Requirement Document	<p>Shaohu</p> <ul style="list-style-type: none"> <li>a) Introduction</li> <li>b) General description</li> <li>c) Log of meeting</li> <li>d) Change of control</li> <li>e) Review</li> </ul> <p>Appala</p> <ul style="list-style-type: none"> <li>f) Functional analysis</li> </ul> <p>Hussein</p> <ul style="list-style-type: none"> <li>g) Nonfunctional analysis</li> </ul>	9/22/2016
Design Specification	<ul style="list-style-type: none"> <li>a) System Design;</li> <li>b) System architectural design.</li> </ul>	10/6/2016
Acceptance Test Plan	<ul style="list-style-type: none"> <li>a) Acceptance test plan;</li> <li>b) Test technical.</li> </ul>	11/10/2016
System Test Plan	<ul style="list-style-type: none"> <li>a) System test plan;</li> <li>b) System test technical.</li> </ul>	11/17/2016
User's Guide	<ul style="list-style-type: none"> <li>a) Installation;</li> <li>b) Guide documentation</li> </ul>	12/01/2016

### **3.7 Data Management Plan**

E-Climbing will use MySQL database application to manage the historical sales, customer information and check-in. The database will include and maintain five primary tables:

- Employees
- Customer information
- Sales
- Check-in
- Reservation

The element of data management includes:

- Data description

A description of the information to be gathered, and the nature and scale of the data that will be generated or collected.

- Data format

Formats in which the data will be generated, maintained and made available.

- Data security and backup

The data will be stored in an external storage server in SDSU with the capacity of 10,000 terabytes. A data backup operation will be running at 2 :00 am Pacific time. All zipped backup data will be uploaded to SDSU Cloud server.

### **3.8 Risk Management Plan**

Variety of possible risk and the action that will take are shown in below table.

#	Risk item	Probability (1~5)	Impact (1~5)	Priority (P*I)	Action	who	Cost
1	Failure to deliver the product on time	3	5	15	Strictly follow WBS and CPA	Shaohu	\$28K
2	Failure to estimate the cost	3	5	15	Review and estimate it again	Hussain	\$28K
3	Failure to achieve the requirement of client	2	4	8	Review client requirements and apply to it	Appala	\$15K
4	Employee turnover	1	2	2	Hire a new employee	Shaohu	\$3K

5	Ineffective communications	1	2	2	Weekly communication	Hussain	\$3K
6	Software freeze	2	5	10	Debug and test the code	Appala	\$19K
7	Response time too long	1	4	4	Code and system optimization	Shaohu	\$7K
8	Bugs	3	3	9	Debug it and get it fixed	Hussain	\$17K
9	Glitch after update	3	3	9	Find it and fix it	Appala	\$17K
10	Fail backup	1	5	5	Set up a new backup	Shaohu	\$9K
11	Insecure database	1	5	5	Encrypt data base	Hussain	\$9K
12	Not encrypted payment details	1	5	5	Rencrypt the payment details	Appala	\$9K
13	Duplication of random ID	1	3	3	Check the random number function	Shaohu	\$5K
14	No on time notification	2	3	6	Fix notification function	Hussain	\$11K
15	Software not able to work on other tablet devices	2	3	6	Find the problem and fix it to work on all tablets devices	Appala	\$11K

### 3.9 Test Plan

Test Plan consists of four levels listed below:

- **Integration Testing**

We do this test after we test each unit and ensure that fulfilled the requirement after we combine them together , each function is represented as a unit.

- **Qualification Test**

The Qualification Test is to ensure quality following points:

- **Correct functionality**
- **Security**
- **Performance**
- **Reliability**

- **System Testing**

In this case our team will test a complete software and hardware, after integrated the system to evaluate the system's compliance with its specified requirements.

- **Acceptance Testing**

This will be the documents specifies what will be tested for the customer to accept. If all listed testing item have being met the client requirement and the client has approved to accept it .

### **3.10 Security Plan**

The purpose of the system security plan (SSP) is to provide an overview of the security requirements of the system and describe the controls in place or planned, responsibilities and expected behavior of all individuals who access the system.

Security strategy:

- Security attributes of the system.
- Restriction the access to features.
- Protection the privacy of the users.

Security Testing process.

- System Architecture
- Analyse System Boundaries.
- Analyse System Access Policies.
- Test Environment.
- Threat Modelling & Attack Vectors.
- Impact/Risk Analysis
- Bug Fixing.

### **3.11 Deliverables**

List of deliverables

● Proposal	9/15/2016
● Estimates	9/15/2016
● Requirement document	9/22/2016
● Design document	10/6/2016
● System test plan	11/10/2016
● Acceptance test plan	11/10/2016
● Final project report and source code	12/1/2016
● User guide	12/2/2016

## 4. ESTIMATES

### 4.1 Estimate Size and Cost

Constructive Cost Model (COCOMO) is a software cost estimation method based on a set of empirically derived equations. COCOMO is proved by a study of 63 projects at TRW Aerospace. The study examined projects ranging in size from 2,000 to 100,000 lines of code. This model is very matched our project feature. We used COCOMO 81 to estimate the size and cost.

COCOMO models include 3 software development types:

- Organic: relatively small software teams develop familiar types of software in an in-house environment. Most personnel have previous experience on similar system in the organization
- Embedded: the project may require new technology, unfamiliar algorithms, or an innovative new method for solving the problem
- Semi-detached: having a mixture between organic and embedded types

#### Basic COCOMO

Type	Effort	Schedule
Organic	PM=2.4 (KDSI) <sup>1.05</sup>	TD=2.5 (PM) <sup>0.38</sup>
Semi-detached	PM=3.0 (KDSI) <sup>1.12</sup>	TD=2.5 (PM) <sup>0.35</sup>
Embedded	PM=3.6 (KDSI) <sup>1.20</sup>	TD=2.5 (PM) <sup>0.32</sup>

Where,

PM = person-month (man-month)

KDSI = delivered source instructions, in thousands

TD = number of months estimated for software development

Our system belongs to Organic type. The below table gives the estimation of KDSI. The cost is \$ 500 per person-day.

#### Basic COCOMO Calculation

This project uses an organic type project with an estimated size = 7,500 lines of code.

WBS	# of code line	cost
1 home	100	\$3,209
1.1 Sign in	200	\$6,643
1.1.1 Registration form	200	\$6,643

1.1.2 log in	200	\$6,643
1.1.1.1 wavier form	200	\$6,643
1.1.1.2 validate user	200	\$6,643
1.1.1.3 Payment method	200	\$6,643
1.2 inventory	400	\$13,755
1.2.1 Equipment data	200	\$6,643
1.2.1.1 notification of the equipment	200	\$6,643
1.2.2.1 levels of climbing	200	\$6,643
1.1.2.1 Patron	200	\$6,643
1.1.2.2 Employee	200	\$6,643
1.1.2.3 Adminstrater	300	\$10,169
1.1.2.2.1 instructor	200	\$6,643
1.1.2.2.2 staff	200	\$6,643
1.1.2.2.1.1 Register class	200	\$6,643
1.1.2.2.1.2 EMS	200	\$6,643
1.1.2.2.1.3 Certification	200	\$6,643
1.1.2.2.1.2.1 Schedule	200	\$6,643
1.1.2.2.1.2.2 Send notifications	200	\$6,643
1.1.2.2.2.2 Check in	300	\$10,169
1.1.2.2.2.3 Send notifications	200	\$6,643
1.1.2.2.2.4 Reservation	200	\$6,643
1.1.2.1.1 Update data	200	\$6,643
1.1.2.1.2 Account	200	\$6,643

1.1.2.2.2.7 Daily note	200	\$6,643
1.1.2.3.1 reservation	400	\$13,755
1.1.2.3.2 EMS	200	\$6,643
1.1.2.3.3 Approve suspend user	200	\$6,643
1.1.2.3.4 Manage staff	200	\$6,643
1.1.2.3.5 Generate report	400	\$13,755
1.1.2.3.4.1 Add/delete staff	200	\$6,643
1.1.2.3.4.2 Update staff data	200	\$6,643
Total	7500	\$250,824

The total cost is \$250,824 based on COCOMO model. After discount, our company proposal the amount cost as \$200,000.

## 4.2 Benefit

We estimate that Wall Climbing Center using this system will be able to increase profits by 10% based on market research we have done.

## 5. TERMS OF ACCEPTANCE

Nester Software shall use commercially reasonable efforts to deliver to end user, no later than 7 days after the Effective Date OR as soon as commercially practicable. For each Deliverable under this Agreement, End user shall have a 14 day "Acceptance Period" beginning on the Delivery Date. During the Acceptance Period, Customer may cancel the license by giving written notice to Vendor and returning the Deliverable in a commercially reasonable manner. Unless such cancellation notice is given, the license will be deemed accepted by Customer at the end of the Acceptance Period

## 6. TERMS AND CONDITIONS

- General

This Agreement, upon acceptance by you (the “End User”), which acceptance shall be indicated by signature or other recorded End User confirmation, forms a legally binding agreement between the End User and Nester

This Agreement shall take precedence over any terms and conditions sought to be relied upon by the End User in respect of the Online Service. Notwithstanding any language on any correspondence received from the End User to the contrary, this Agreement shall take precedence over any such correspondence. Such correspondence shall be accepted by Nester for administrative purposes only and shall not modify or amend this Agreement. All terms and conditions on any correspondence originating from the End User shall be null, void and without legal effect.

- **Grant of License to End User**

- a) License grants are subject to the End User’s obligation to pay and continue paying the Subscription Charges and the End User’s compliance with this Agreement and any additional product use terms associated with this Agreement.
- b) Limitations on use. Licensed software is licensed to the End User, not sold. The End User has no right to:
  - i. reverse engineer, decompile, or disassemble any licensed software, except where applicable law permits it despite this limitation;
  - ii. rent, lease, lend, resell, or host to or for third parties any licensed software, except as may be expressly permitted for a given licensed software in the product use terms.

## **7. TERMS OF PAYMENT**

All prices are quoted exclusive of all taxes and transportation charges unless otherwise agreed by both parties. Payment terms are money order or check in advance, unless otherwise stated. The buyer shall be in default of payment without reminder. In the event of non-payment, Nester Software reserves the right to alter terms of payment, suspend credit and delay shipment and pursue any remedies available at law or under this agreement.

**The payment shall be broken down into sections which are as follows:**

- 30% on down payment upon receipt of the confirmation of order and signing the contract
- 10% on successful system acceptance testing
- 10% on successful user acceptance testing

- 10% on successful ‘live running’ or upon delivery
- 40% due 30 days after delivery of product

Each party represents and warrants that on this date they are duly authorized to bind their respective principals by their signatures below.

**Customer:**

---

(Signature)

---

(Typed or Printed Name)

Title: \_\_\_\_\_

Date: \_\_\_\_\_

**Developer:**

---

(Signature)

---

(Typed or Printed Name)

Title: \_\_\_\_\_

Date: \_\_\_\_\_

## **8. WARRANTY**

The software furnished under this agreement is provided on an “as is” basis, without any warranties or representations express, implied or statutory; including, without limitation, warranties of quality, performance, non-infringement, merchantability or fitness for a particular purpose. nor are there any warranties created by a course of deadline, course of performance or trade usage. Developer does not warrant that the software will meet customer’s needs or be free from errors, or that the operation of the software will be uninterrupted. The foregoing exclusions and disclaimers are an essential part of this agreement and formed the basis for determining the price charged for the software.

The warranties set forth in this agreement are the only warranties granted by developer. Developer disclaims all other warranties express or implied, including , but not limited to, any implied warranties of merchantability or fitness for a particular purpose.

## **9. APPENDIX**

Log of Meetings and Review

8/25 5-6PM group meeting WENS Lab 1hr

- 1) Startup
- 2) Discuss the IDE and programming language

Java

8/27 9-11PM group meeting WENS Lab 1hr

- 1) Discuss the big picture
- 2) Requirement and detail
- 3) Question about client

4) Discuss the company name and logo

Nester Software, Inc

8/28 9pm -12:30am Shaohu Review 3.5 hrs

Logo design

9/4 10am -11am Tue group meeting WENS Lab 1 hr

- 1) Split work

9/6 9-12:30am 2-5pm Shaohu Review 3.5 hrs

Proposal

Table of content

Working on front page

Document plan

Data management

9/5 6-11PM Mon group meeting WENS Lab 5 hrs

Discuss the CPA

9/6 9-11PM Tue group meeting WENS lab 2hrs

- 1) Activity Gantt chart
- 2) Secure plan

9/9 9-11:59AM Fri group meeting WENS lab 3 hrs

Scope complete

9/14 9-11:59AM Wed group meeting WENS lab 3hrs

- 1) Proposal Discussion
- 2) Proposal complete

## **Request of proposal**

**Wall Climbing Center,  
South Dakota State University,  
Brookings, SD 57006  
Date: October 16, 2016**

**Dear Sir/Madam,**

**We are delighted to inform you that your proposal titled e-Climbing System has been approved by SDSU. We would like to move forward to the contract and next stage.**

**We would ask you to provide the detail requirement documentation including all functionality requirement and description. This would give us a better understanding of project structure and resource you require. Please send us the documentation before 10/22/2016.**

**If you have any question, please feel free to contact us at [Justin.Park@sdstate.edu](mailto:Justin.Park@sdstate.edu)**

**Yours Sincerely,  
Justin Parks**

**Director of Wall Climbing Center**

# **Requirement Document**

## **e-Climbing System**

**Prepared for:**  
The Wellness Center Wall Climbing  
of  
South Dakota State University



**Version 2.0**

**Prepared By:**  
**Nester Software. Inc**

**1400 8th St. Nester Center  
Brookings, SD 57006  
10/2016**

**Approved for public release; distribution is unlimited**

**SOFTWARE DEVELOPMENT REQUIREMENT DOCUMENTATION  
FOR  
E-CLIMBING SYSTEM**

**Prepared By:**  
**Nester Software. Inc**  
**1800 8th St. Nester Center**  
**Brookings, SD 57006**  
**10/2016**

**Team Information**

*Appala Chekuri*

---

**Software Project Manager** **Signature**

*Hussein Otudi*

---

**Systems Engineer Manager:** **Signature**

*Shaohu Zhang*

---

**Chief Executive Officer** **Signature**

Date	Description	Revision	Editor
9/10/2016	Created the document	0	Shaohu
9/15/2016	Added Table of Content Split work	0.1	Shaohu, Hussein Appala
9/19/2016	Added introduction	0.2	Shaohu
9/20/2016	Added general description	0.3	Shaohu, Hussein
9/21/2016	Polish general description	0.4	Shaohu
9/21/2016	Added WBS	0.5	Appala
9/22/2016	RD draft	0.6	Hussein, Appala Shaohu
10/2/2016	UI design	0.7	Hussein, Appala Shaohu
10/2/2016	Review	0.8	Shaohu
10/3/2016	Formatting, Reviewer	0.9	Shaohu
10/3/2016	Presentation Reviewer	1.0	Hussein,Appala, Shaohu
10/5/2016	UI redesign	1.1	Hussein,Appala, Shaohu
10/7/2016	Delete EMS	1.2	Hussein,Appala, Shaohu
10/10/2016	Update	1.3	Hussein,Appala, Shaohu
10/14/2016	Added revised WBS	1.4	Appala, Hussein
10/15/2016	Added revised report format	1.5	Appala, Hussein
10/16/2016	Final version	2.0	Shaohu,Appala,hussain.

**This deliverable has two hard copies. One is for Dr. Shin from South Dakota State University, and another one is submitted to Mr. Park for approval. Distribution is unlimited for public release.**

## **CHAPTER 2. REQUIREMENT DOCUMENTATION**

### **1. INTRODUCTION**

#### **1.1 Purpose**

This document sets forth the requirements for the “e-Climbing” software application to be developed for SDSU Wall Climbing Center by Nester Software, Inc. (Nester). Both functional and nonfunctional requirements for e-Climbing have been developed through numerous meetings between Mr. Parks and e-Climbing development team. These meetings continue to the present day as the understanding of the requirements set continues to evolve for both parties once Mr. Parks approves the Requirement Document.

This document discusses the purpose and contents of a requirements document for the e-Climbing software application. Client-oriented requirements describe the system from the client’s perspective. These requirements include a description of the different types of users served by the system. Developer-oriented requirements illustrate the system from a software developer’s perspective including a detailed description of functional, data, performance, and other important requirements.

#### **1.2 Objective**

The current system in Wall Climbing Center (WCC) has the following issues and challenges:

- Patron/employee accessibility**

There is no online access system for customer and employees. The customers have to call the program manager for reservation. The available time slot is not accessible for public. Program manager is the only person for dealing with reservation and registration. The class schedule and flyer only can be grabbed in WCC.

- Lack of patron evolvement**

The current system only have employee and admin domain. Patron is unable to register or make reservation online.

- Automatic report generation**

The center uses Excel and EMS for the main tool for daily operation and management. It is very inefficient. For example, the staff has to check register form or visitor log one by one to get weekly or monthly report. It is really time-consuming.

Our goal is not only to solve these issues as much as possible, but also will develop an intelligent and efficient wall climbing management system. These goals include:

- **Patron/employee accessibility**

The system will include more accessibility for Patron and employee.

- **Online reservation**

Users can make reservation on internet by computer, tablet or smartphone.

- **Online waiver**

The system should have ability to accept online waiver. The complete waiver form will be automatically saved in department driver.

- **Automatic report generation**

The system will have fully ability to generate a wide array of report based upon facility access such as user information, participation, new users and so on.

- **Suspension management**

Employees have ability to request suspension of Climbing Wall privileges for individuals that broke policy with an included incident report to describe why the request is being submitted.

Admin have ability to suspend access to the Climbing Wall based upon the incident report generated by the staff member.

System will deny access until the date that the suspension is over.

System will automatically increase suspension by a certain percentage if the individual is a repeat offender.

- **Certification management**

The system tracks whether or not the individual has a current belay or lead climb certification with the climbing gym. The system gives warning of retest if the individual has not checked in at the wall for a period of over three months.

### 1.3 Definitions

Table 1. Definition

Terms	Definition. Acronym, Abbreviation
Code line	Source code required to produce software
DB	Database.
HDD	Hard Disk Drive
Java	A programming languages created by Oracle. We will be using this language to build the e-Climbing.
MS	Microsoft. Microsoft is a large software company which produces the software that will be used to implement ATPS.

Microsoft Access	A database software created by Microsoft. The campus police vehicle violation database was created using Microsoft Access.
SRD	Software Requirement Documentation
SVC	Software Version Control
PC	Personal Computer.
PDA	Personal Digital Assistant. This is a handheld computer similar to a personal computer.
UI	User interface
WCC	Wall Climbing Center

## 1.4 System Overview

The below figure demonstrates the system overview.

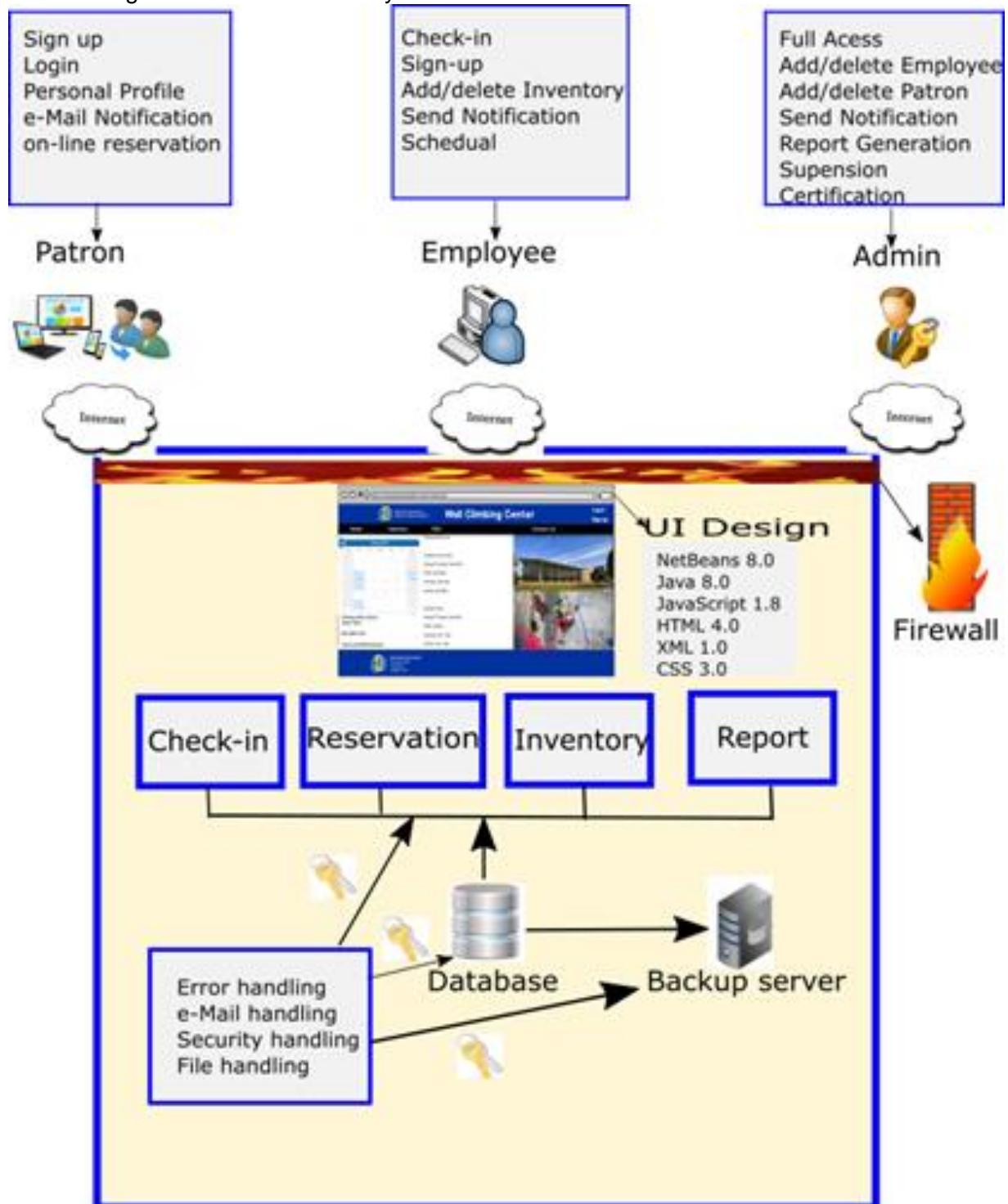


Figure 1. Overview of e-Climbing System

This project has four main systems check in, reservation, Inventory, Pos. all the systems are connected with the database using the backup server with localhost running to connect the dynamic change in the data and access in the data.

## 1.5 Reference

- [1] Software Requirements Document. <http://web.mit.edu/ssit/cis/CISRequirements.html>
- [2] Berezin, Tanya. "Writing a Software Requirements Document." (1999).  
[http://www.ele.uri.edu/Courses/ele205/ELE\\_205/Stuff\\_You\\_Need\\_files/ReqsDoc.pdf](http://www.ele.uri.edu/Courses/ele205/ELE_205/Stuff_You_Need_files/ReqsDoc.pdf)
- [3] South Dakota State University Color  
<https://www.sdsstate.edu/university-marketing-communications/graphic-identity-standards/university-colors>
- [4] Software Requirements Specification  
<http://cohesion.chrysosome.net/documentation/srs/>

## 1.6 Overview of document

The requirement document is organized as follows:

Section 1 presents a brief overview of e-Climbing and the issues driving its desire for development of the application.

Section 2 discusses the product perspective, product functions and user personas.

Section 3 illustrates functional and nonfunctional requirements for the application.

Section 4 is the log of meetings and review.

Section 5 gives the change of control

Appendix lists the FWBS and Activity gantt chart.

## 2. GENERAL DESCRIPTION

### 2.1 Product Perspective

The e-Climbing system includes:

- The basic ability is to maintain the daily operation including check-in, register, reservation and online waiver.
- The inventory system can add, delete and update the item. The summary of items can be generated to providing the sale, storage and profit information.
- The POS system allows for cash register, receipt printer, customer display and debit/credit card reader

### 2.2 Product Functions

The overall description of e-Climbing functionality includes:

1.1 sign in

    1.1.1 registration form

        1.1.1.1 waiver form

- 1.1.1.2 Patron Email validation
- 1.1.1.3 payment method
- 1.1.2 log in
  - 1.1.2.1 patron
  - 1.1.2.2 employee
    - 1.1.2.2.1 register class
    - 1.1.2.2.1.2 EMS
      - 1.1.2.2.1.2.1 schedule class
      - 1.1.2.2.1.2.2 send notification
    - 1.1.2.2.1.3 certification management
    - 1.1.2.2.2 check in management
    - 1.1.2.2.5 request admin for user suspension
    - 1.1.2.2.6 Update Certification
    - 1.1.2.2.7 Daily notes
  - 1.1.2.3 admin
    - 1.1.2.3.1 staff management
    - 1.1.2.3.2 Suspend management
    - 1.1.2.3.3 Reservation
    - 1.1.2.3.4 EMS
    - 1.1.2.3.5 Listserv
    - 1.1.2.3.6 report generation
- 1.1.3 support desk info
- 1.2 inventory
  - 1.2.1 equipment data
  - 1.2.2 certification
  - 1.2.3 search
- 1.3 POS
  - 1.2.1 equipment data
  - 1.2.2 certification
  - 1.2.3 search

## 2.3 User Characteristics

There are three user characters including admin, employee and patron.

- Admin

Someone needs to be responsible for security and maintenance of the system. This is the System Administrators role. The administer of system has complete access.

- Add/delete employees and patrons.
- Add/delete inventory.
- Create calendar reservations.
- Ability to suspend access to the Climbing wall.
- Employees

These are the most obvious users. Employee should be familiar with the system functionality and UI.

- can add/delete inventory;
- can create calendar reservations
- Patron

More often than not, software is designed for a client. The client may wish to see the design as laid out by the system and be able to see what exactly they are buying. Patron can

- Create and update personal profile
- Check the available schedule
- Purchase item

## 2.4 Design and Implementation Constraints

- **Color**

The e-climbing webpage color follows the consistent use of South Dakota State University's official colors for the purpose of building brand consistency and awareness in the marketplace. The primary visual identifier of webpage is blue as shown in Figure 2.



Figure 2. Primary colors

The secondary color should comply with the university's graphic standards as shown in Figure 3. This range of colors may be used in limited quantities such as highlight text or graphs, but they should never appear as the dominant color in any webpage.

PMS 1795 0c 96m 90y 2k 210r 38g 48b #D22630	PMS 185 0c 92m 76y 2k 228r 0g 43b #E4002B	PMS 186 (PRT) 0c 100m 75y 4k 200r 16g 46b #C8102E	PMS 200 (USD) 3c 100m 66y 12k 186r 12g 47b #BA0C2F	PMS 416 22c 14m 24y 45k 126r 127g 116b #7E7F74
PMS 430 33c 18m 13y 37k 124r 135g 142b #7C878E	PMS 444 38c 15m 18y 43k 113r 124g 125b #717C7D	PMS 542 64c 19m 1y 4k 123r 175g 212b #7BAFD4	PMS 549 59c 8m 9y 19k 107r 164g 184b #6BA48B	PMS 5493 46c 5m 14y 14k 127r 169g 174b #7FA9AE
PMS 123 0c 28m 98y 0k 255r 119g 44b #FFC72C	PMS 012 0c 8m 98y 0k 225r 215g 0b #FFD700	PMS 118 5c 26m 100y 27k 172r 132g 0b #AC8400	PMS 137 0c 38m 95y 0k 255r 163g 0b #FFA300	PMS 110 2c 24m 100y 7k 218r 170g 0b #DAAA00
PMS 335 100c 0m 58y 22k 0r 123g 95b #007B5F	PMS 347 (4-H) 89c 0m 90y 0k 51r 153g 102b #339966	PMS 348 100c 4m 87y 18k 0r 132g 61b #00843D	PMS 349 94c 11m 84y 43k 4r 106g 56b #046A38	PMS 359 42c 0m 48y 0k 161r 216g 132b #A1D884
PMS 376 53c 0m 96y 0k 132r 189g 0b #84BD00	PMS 377 51c 5m 98y 23k 122r 154g 1b #7A9A01	PMS 383 26c 3m 93y 17k 168r 173g 0b #A8AD00	PMS 389 23c 0m 83y 0k 208r 223g 0b #D0DF00	PMS 575 57c 11m 85y 45k 103r 130g 58b #67823A
PMS 7494 31c 5m 36y 16k 156r 175g 136b #9CAF88	PMS 462 28c 48m 71y 72k 92r 70g 43b #5C462B	PMS 464 13c 51m 87y 48k 139r 91g 41b #8B5B29	PMS 471 5c 70m 97y 20k 184r 97g 37b #B8E125	PMS 513 56c 98m 0y 0k 147r 50g 142b #93328E
PMS 520 66c 86m 5y 17k 100r 47g 108b #642F6C	PMS 525 71c 90m 9y 37k 87r 44g 95b #572C5F	PMS 526 76c 99m 0y 0k 112r 47g 138b #702F8A	PMS 527 75c 100m 0y 0k 128r 49g 167b #8031A7	

Figure 3. Secondary color

- Operating system

**Table 2. 64-Bit e-Climbing Operating System Requirement**

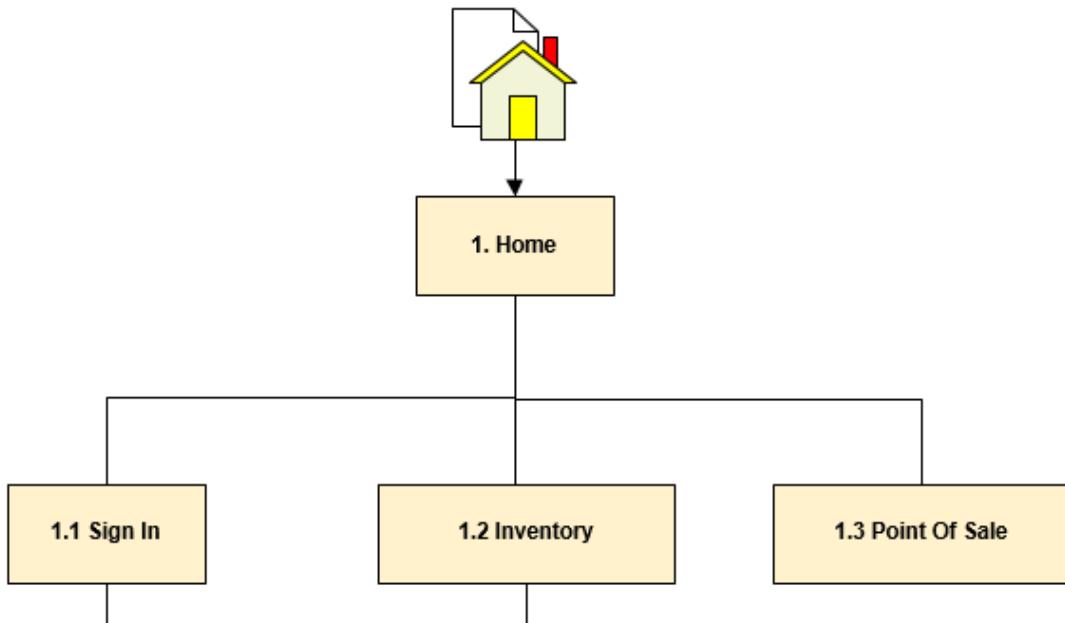
Operating Systems	Processors	Disk Space	RAM	Graphics
Windows 10	Any Intel or AMD x86-64 processor	1 GB for e-Climbing only, 2–4 GB for a typical installation	4 GB is required	No specific graphics card is required. Hardware accelerated graphics card supporting OpenGL 3.3 with 1GB GPU memory is recommended.

### 3. SOFTWARE REQUIREMENT

#### 3.1 Functional Requirements

Functional requirements are demonstrated by FWBS, flow chart, UI screenshot and table in this section.

1) Homepage:



**Description:** Home page (1) contains the start screen for the application.

**Input:** login (1.1.2), sign up(1.1.), Inventory(1.2) and POS(1.3) link button.

**Output:** go corresponding linked website.

The screenshot shows the homepage of the Wall Climbing Center at South Dakota State University. At the top, there is a navigation bar with links for Home, Inventory, POS, Checkin, Contact Us, Log in, and Sign up. Below the navigation bar is a banner featuring the university's logo and the text "Wall Climbing Center". To the left, there is a calendar for October 2016. In the center, there is information about climbing wall hours, academic year hours, summer hours, and climbing wall contact details. On the right, there is a photograph of two people climbing on an outdoor rock wall. At the bottom, there is a footer with the university's logo and contact information.

Screen 1.1

### 1.1) Login page:

**Description:** The login page will have the username and password field to enter the system.

**Input:** username(see Table 3,Screen 1.1) ;  
password(see Table 3,Screen 1.1).

**Output:** It enters the page of user if the user is registered,  
otherwise ask for registration

<https://www.sdsstate.edu/wellness-center/climbing-wall>

**Wall Climbing Center**

Email

Password

[Home](#)

**South Dakota State University**  
Brookings, SD 57007  
1.800.952.3541  
Copyright © 2016

Screen 1.2

<https://www.sdsstate.edu/wellness-center/climbing-wall>

**Wall Climbing Center**

Username

Password

Your Password or Username is not correct

[Home](#)

**South Dakota State University**  
Brookings, SD 57007  
1.800.952.3541  
Copyright © 2016

Screen 1.3

Table 3. Log in format

Number	Field	Data Type	Requirement	Example
1	Email	String	<u>xxx.xxx@xxx.xxx</u> xxxx@xxxx.xxx Case sensitive Length<45	shaohu2016@gmail.com shaohu.zhang@sdsstate.edu
2	Password	String	Characters, _ or digit 5<Length<25	12frgh89

### 1.1.1) Registration form:

**Description:** Allows user to enter the information and validate the information.

**Input:** The input of the user will be as described in the table.(see screen 1.4)

**Output:** Successfully validate the user and update to the database.

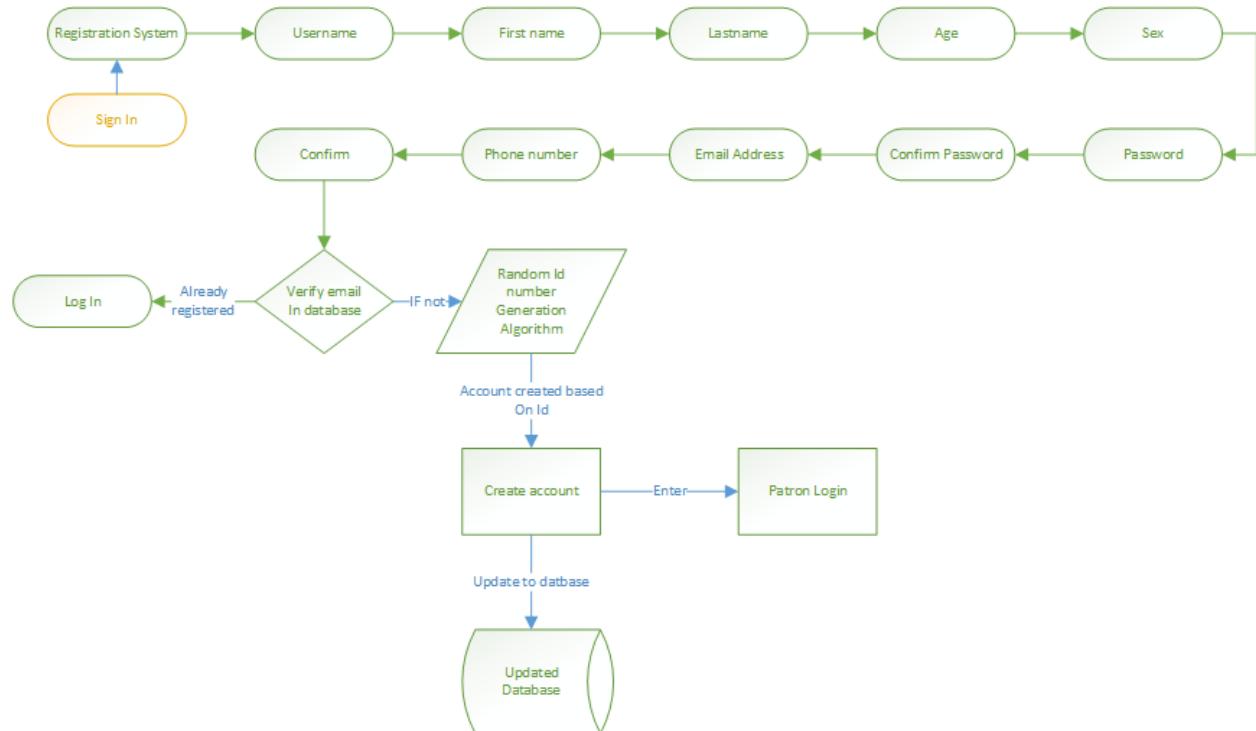
Screen1.4

Table 4. Register form

Number	Field	Data Type	Requirement	Example
1	First name	String	Characters: Uppercase or Lowercase 1<Length<25	Shaohu shaohu
2	Lastname	String	Characters: Uppercase or Lowercase 1<Length<25	Zhang zhang

3	Password	String	Characters,_ or digit 5<Length<25	12frgh89
4	Conform Password	String	Characters,_ or digit 5<Length<25	12frgh89
5	Email	String	<u>xxx.xxx@xxx.xxx</u> xxxx@xxxx.xxx Case sensitive Length<45	shaohu2016@gmail.com shaohu.zhang@sdstate.edu
6	Email	String	<u>xxx.xxx@xxx.xxx</u> xxxx@xxxx.xxx Case sensitive Length<45	shaohu2016@gmail.com shaohu.zhang@sdstate.edu
7	Phone Number	Digit	Format: XXX-XXX-XXXX	605-592-0488
8	User Name	String	Characters: Uppercase or Lowercase 5<Length<25	Shaohu23

### Flow chart to represent Registration form:

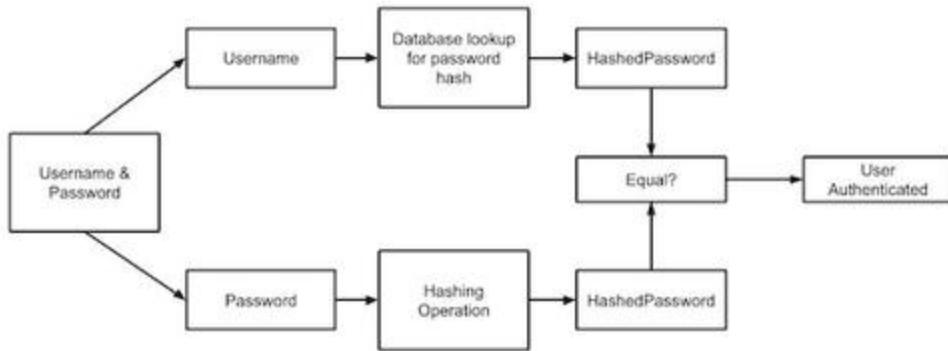


#### 1.1.1.a) Encrypt Password:

**Description:** Encrypt the password for the user authentication with the hashed function.  
**Implementation:**

**Input:** Takes the password as the input for the hashing algorithm.

**Output:** Creates a encrypted hash functional output for user authentication.



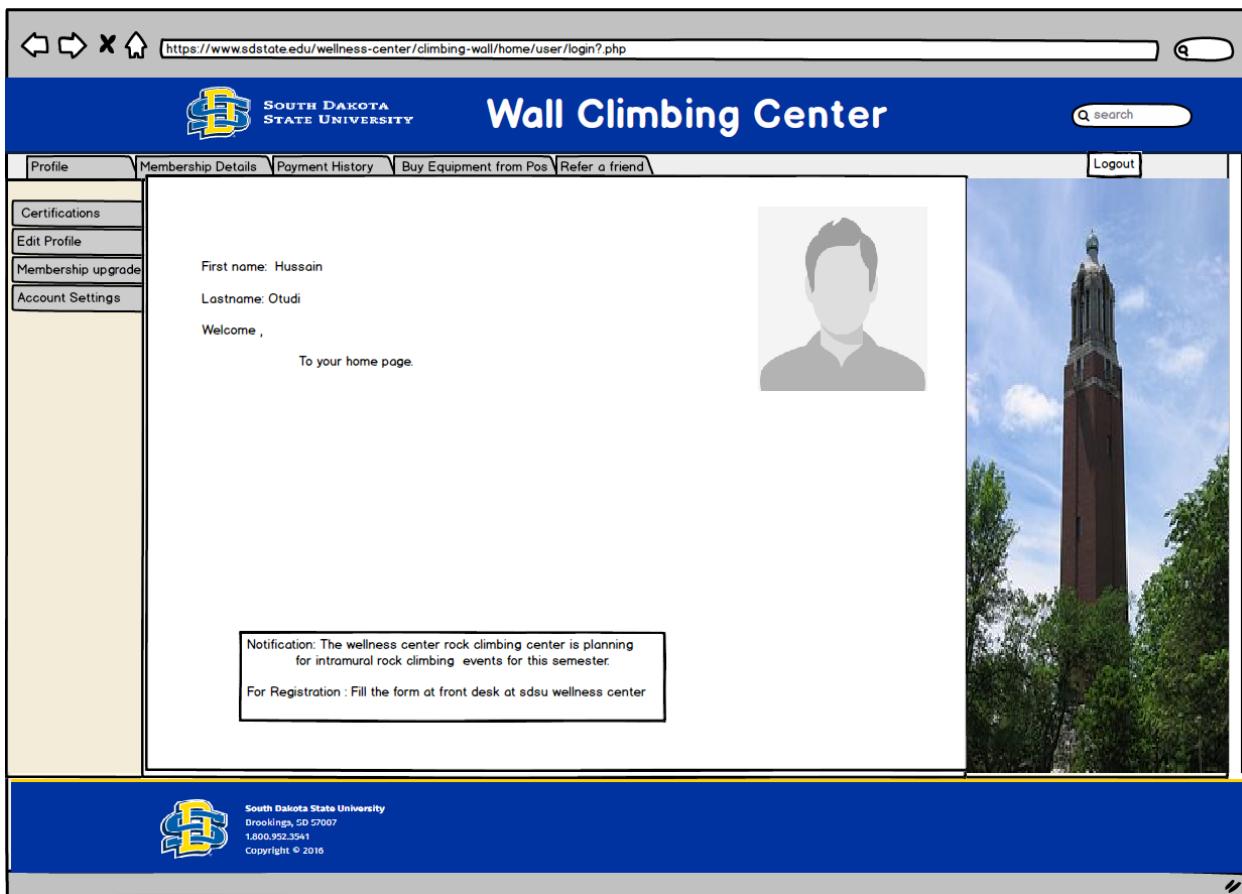
#### 1.1.1.b) Confirmation Email:

**Description:** This algorithm will check for the user email address for the duplicate account. If the account has already there then it shows the message email already exists.

**Input:** email address.

**Output:** make sure no account is duplicated.

```
public void EmailTests(string email, bool expected)
{
    string pattern = @"^(?!.).(([^"\r\\"]|\\[""\r\\"])*""|"
        + @"([-a-z0-9!#$%&'*/=?^`{|}~](?<!\\.\\.)*)?(?<!\\.\\.)"
        + @"@[a-z0-9][\w\.-]*[a-z0-9]\.[a-z][a-z\.]*[a-z]$";
    Regex regex = new Regex(pattern, RegexOptions.IgnoreCase);
    Assert.AreEqual(expected, regex.IsMatch(email)
        , "Problem with " + email + ". Expected "
        + expected + " but was not that.");
}
```



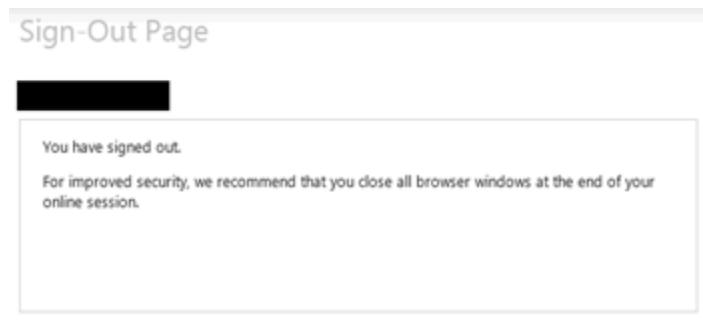
Screenshot 1.5

#### 1.1.1.c) logout:

**Description:** Allows the user to sign out of their account

**Input:** User clicks on Sign out.

**Output:** User will get out of their account.(see screen 1.6)



Screenshot 1.6

#### 1.1.1.d) Random Id Generation:

**Description:** This random Id generation algorithm will generate a unique ID for each person who will register for this system.

**Input:** Submit button of the registration form

**Output:** generate a Unique ID for the user to check In every time.

**Algorithm:**

```
counter = 0
function get_new_id()
myid = SHA1(string(counter) + seed)
counter = counter + 1
return myid
End
```

**1.1.1.1) Waiver form:**

**Description:** The Waiver form will describe all the risks in climbing wall and the coverage

of our insurance surety

**Input:** Just fill the form and submit.(see screen 1.7)

**Output:** Validate the form and submit the copy to the database.



### SDSU Climbing Gym Acknowledgement of Risk

**PLEASE NOTE: This Waiver of Liability, Release, Acknowledgement of Risk, and Indemnification Agreement ("Waiver Agreement") is intended to be, and is, legally binding.**

If any aspect of this Waiver Agreement requires clarification, have an SDSU Wellness Center Climbing Gym attendant fully explain it, before signing. By signing the Wellness Center Climbing Gym "Sign-in Sheet(s)", you are agreeing to all terms set forth in this Waiver Agreement. You and/or the person on whose behalf you are signing, are waiving the right to bring any type of action, whether in court or otherwise, to recover compensation or obtain any other remedy for any personal injuries, damages to property, any accident or incident of any type, or death, arising out of or related to your use of the Wellness Center Climbing Gym, its facilities, grounds, climbing walls, exercise areas, equipment, whether the use is supervised or unsupervised. Rock climbing is a sport that has inherent risks. While the SDSU Wellness Center Climbing Gym offers the sport of rock climbing in a controlled environment, there is still an assumed risk of injury to persons using the SDSU Wellness Center Climbing Gym. In agreeing to this Waiver Agreement, I hereby acknowledge, understand, and agree on my behalf, and upon behalf of the person for whom I am signing, that the sport of rock climbing and the use of the SDSU Wellness Center Climbing Gym, its facilities, equipment, climbing walls, classes and/or participating in activities sponsored by the SDSU Wellness Center Climbing Gym has inherent risks. These risks include, but are not limited to any injury or damage resulting from:

Negligence of employees, or the SDSU Wellness Center Climbing Gym. Negligent misuse of the facility, climbing walls, or equipment of the SDSU Wellness Center Climbing Gym; Falling off or impacting against the climbing walls, impact surface, floors, or anything else; Rope abrasion, entanglement or other activities occurring on the premises; Cuts or abrasions resulting from any cause whatsoever; Failure of the climbing walls or equipment, whether inside or outside; Personal health problems, whether mental or physical; Negligence of other climbers, visitors, or observers or persons who may be present in or around the climbing area or facility; and/or Negligence or lack of adequate training or any person(s) who seek to assist with medical or other help either before or after any injury or damage may occur.

By signing the "Sign-in Sheet", I, for myself and for my heirs, next of kin assigns, and personal representatives, hereby agree to and do release, indemnify and hold harmless the State of South Dakota, South Dakota Board of Regents, South Dakota State University, and the SDSU Wellness Center Climbing Gym, and their officers, employees, and agents, and/or volunteer assistants, from any and all injuries and damage which I, or the person upon whose behalf I am signing in, may sustain or incur arising out of or related to my use of the SDSU Wellness Center Climbing Gym, its facilities, grounds, climbing walls, exercise areas, equipment, participation in classes or events, and/or outdoor programs guided by or connected with the SDSU Wellness Center Climbing Gym, whether the use is supervised or unsupervised. I, for myself and for my heirs, next of kin, assigns, personal representatives, and persons upon whose behalf I am signing the "Sign-in Sheet," hereby agree to and release, indemnify and hold harmless the State of South Dakota, South Dakota Board of Regents, South Dakota State University, and the SDSU Wellness Center Climbing Gym, and their agents, employees, offices, and, volunteer assistants, from any and all causes of action, claims for damages or demands whatsoever. **THIS WAIVER AGREEMENT IS BINDING EVEN IF THE RELEASED PERSON(S) OR ENTITY(IES) HAVE CAUSED OR CONTRIBUTED TO ANY DAMAGE OR INJURY THROUGH THEIR COLLECTIVE OR INDIVIDUAL NEGLIGENCE.**

*I and/or person on whose behalf I am signing-in, voluntarily assume complete responsibility for risks and any injuries or damage which may occur as a result of those risks even if the manner or type of injury or damage occurs in a manner that is not foreseeable at the time this Waiver Agreement is accepted. In consideration of my use of the gym, its equipment, employees, volunteer assistants, independent contractors, I agree to and do release, indemnify and hold harmless, the SDSU Wellness Center Climbing Gym, and any and all of their agents, servants and employees, from all liability, claims, demands and damages and further promise not to commence any action or proceeding asserting same.*

All climbers who are twelve (12) years of age or under must be directly supervised by an SDSU Wellness Center Climbing Gym approved adult or be a participant in an SDSU Wellness Center Climbing Gym program. The "Sign-in Sheet" is being signed by the youth's parent, legal guardian, or adult authorized to sign by the youth's parent or legal guardian. By signing the "Sign-in Sheet", the adult acknowledges that they understand the terms of the Waiver Agreement and has the authority to sign for the youth climber. The person signing the "Sign-in Sheet" understands and acknowledges that this Waiver Agreement is binding on the person on whose behalf the "Sign-in Sheet" is signed, for their heirs, next of kin, assigns, and personal representatives.

BY SIGNING the SDSU Wellness Center Climbing Gym "SIGN-IN SHEET" I ACKNOWLEDGE THAT I HAVE READ AND AGREE TO THE TERMS OF THIS WAIVER AGREEMENT. THERE ARE NO ORAL REPRESENTATIONS, STATEMENTS, OR INDUCEMENTS WHICH HAVE BEEN MADE THAT ALTER, CHANGE OR MODIFY ANYTHING SET FORTH IN THIS WAIVER AGREEMENT.

	Print Name	Sign Name	Member Type	Gender: *Optional*	Date & Time Arrived	Time Left
1.						

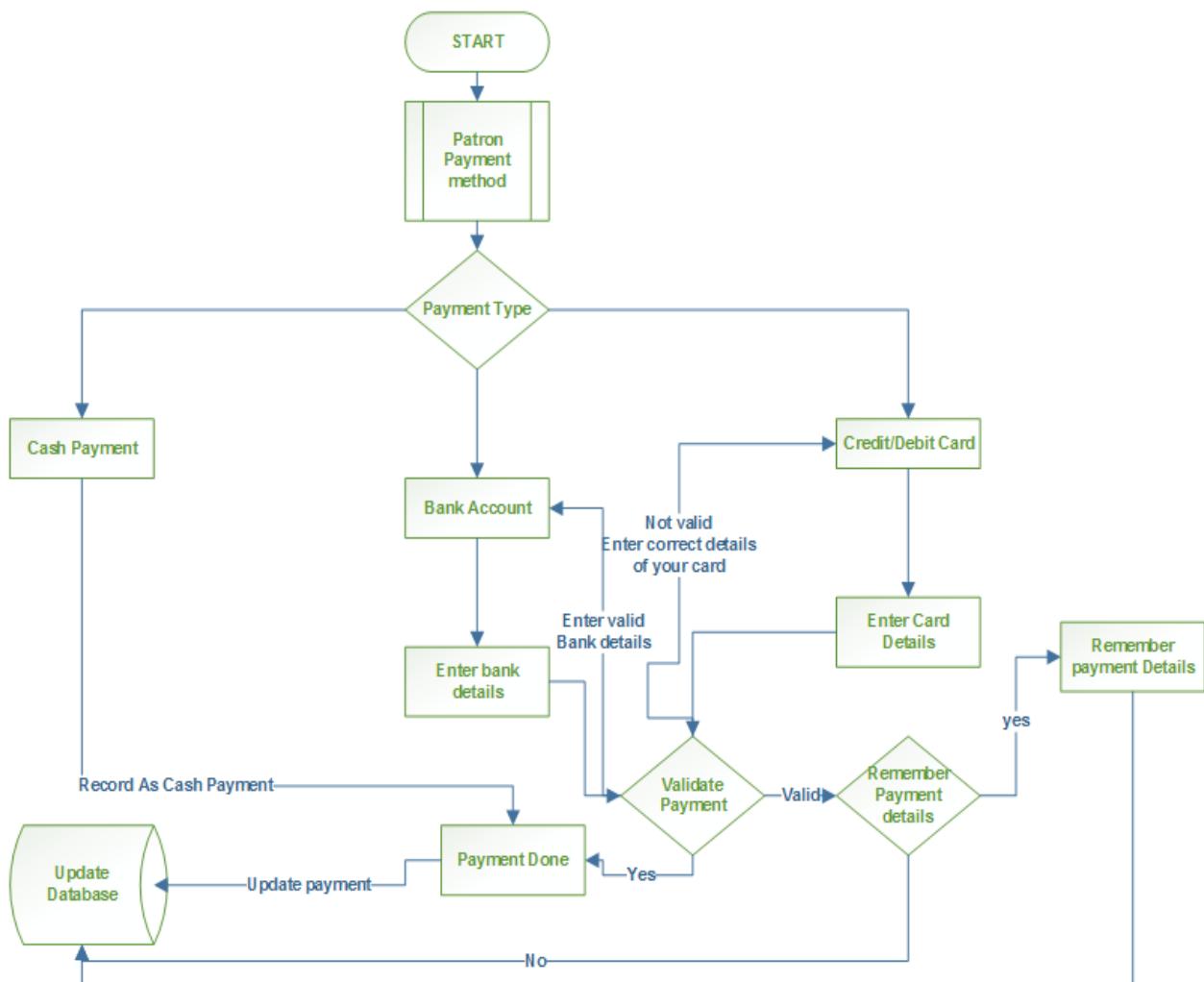
**Screenshot 1.7**

#### 1.1.1.3) Payment method:

**Description:** Payment method can be any of the cash payment, payment with the card details or the bank account

**Input:** Give the information about payment of \$25 for the user at registration(see Sc:1.8)

**Output:** make the payment of \$25 done.



<https://www.sdsstate.edu/wellness-center/climbing-wall/Appala/??/Chekuri/Payment.php?/>



**SOUTH DAKOTA  
STATE UNIVERSITY**

## Wall Climbing Center

**Pay with a debit or credit card**  
If you don't have a PayPal account

Country

Credit Card number

Expiration Date  mm /  CSC  [What's this?](#)

First name

Last name

Billing Address Line 1

Billing Address Line 2 (Optional)

Zip code

City

State

Phone

Email Address



[◀ Make Payment](#)


 South Dakota State University  
 Brookings, SD 57007  
 1.800.952.3541  
 Copyright © 2016

**Screenshot 1.8**

**Table 5. Payment form**

Number	Field	Data Type	Requirement	Example
1	Country	String	Characters: Uppercase or Lowercase 1<Length<20	China
2	Credit number	Digit	XXXX-XXXX-XXXX-XXXX	1234-5678-9112-9999
	CSC	Digit	XXX Length=3	444
3	Expiration Date	Digit	Format mm/dd/yyyy	14/04/1989
4	First Name	String	Characters: Uppercase or Lowercase 1<Length<25	Hussain hussain

5	Last Name	String	Characters: Uppercase or Lowercase 1<Length<25	Otudi otudi
6	Billing Address1	String	Character or digit and 1< length<50	2229 10th St
7	Billing Address2	String	Character or digit and 1< length<50	
8	Zip code	Digit	XXXXX	57006
9	City	String	Characters: Uppercase or Lowercase 1<Length<10	Brookings
10	phone	Digit	Format: XXX-XXX-XXXX	605--592-0069
11	Email Address	string	<u>xxx.xxx@xxx.xxx</u> xxxx@xxxx.xxx Case sensitive Length<45	Otudi.2105@gmail.com Hussain.Otudi@sdstate.edu

#### 1.1.1.3.a) Payment process:

**Description:** This describes the process payment how the payment has been done.

**Input:** Give the details of the payment. (see screenshot 1.8)

**Output:** Submit the payment.

#### 1.1.1.3.b) Update payment info:

**Description:** This will update the payment information to the database

**Input:** The submit button of payment is the input.

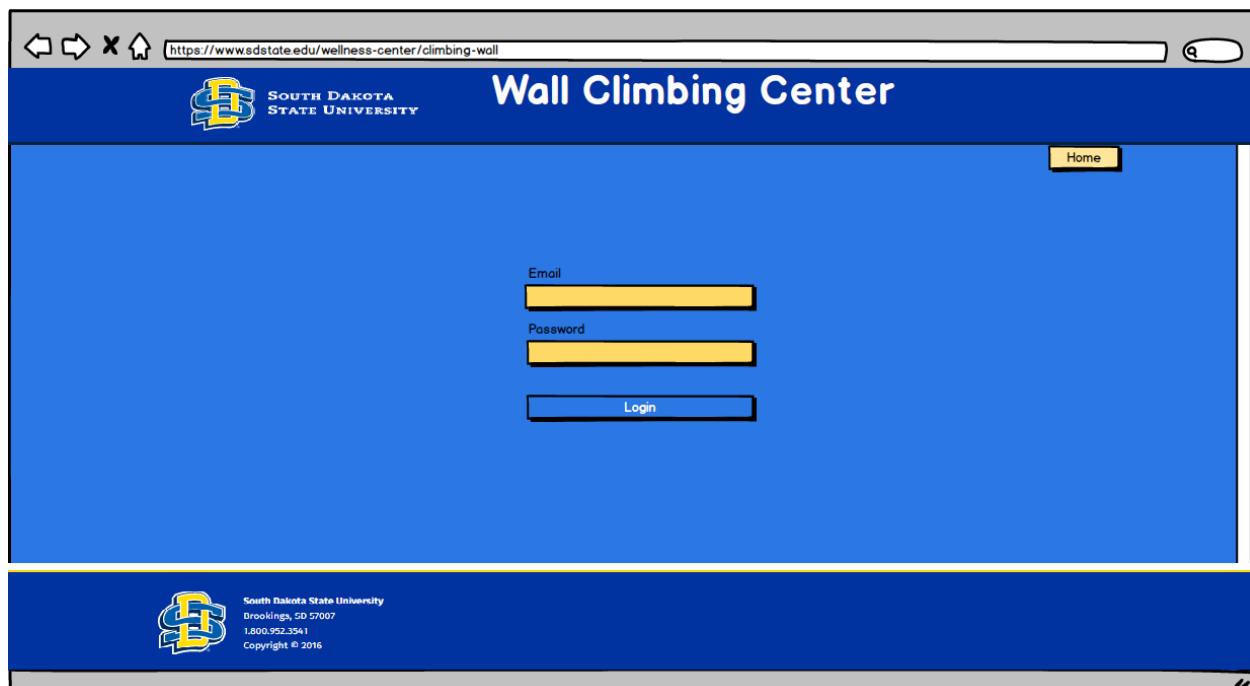
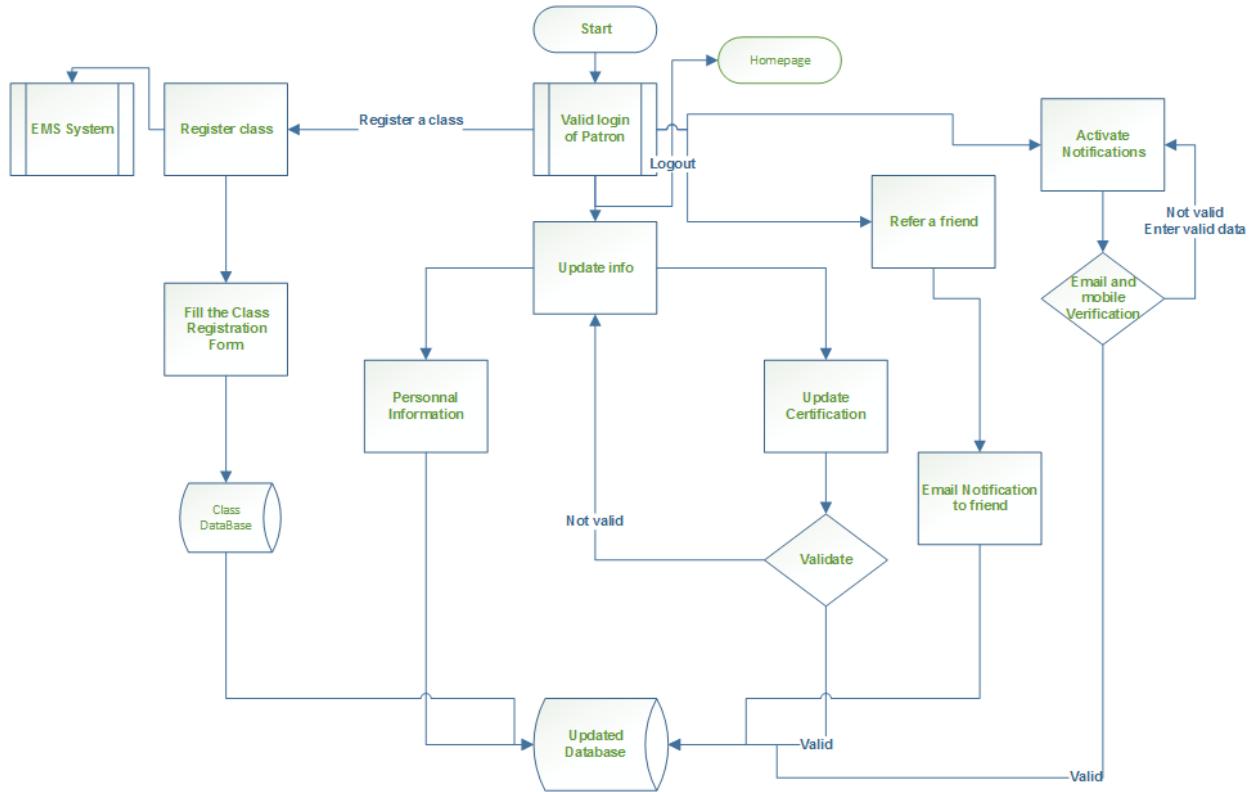
**Output:** The update of payment will be submitted and stored in the database.

#### 1.1.2.1) User login:

**Description:** This validates the user to login to his account.

**Input:** Click the login button after the login fill page. (see screenshot 1.9)

**Output:** Opens the user page. (see screenshot 1.5)



Screen 1.9

Number	Field	Data Type	Requirement	Example
--------	-------	-----------	-------------	---------

1	Email	String	<u>xxx.xxx@xxx.xxx</u> xxxx@xxxx.xxx Case sensitive Length<45	shaohu2016@gmail.com shaohu.zhang@sdstate.edu
2	Password	String	Characters,_ or digit 5<Length<25	12frgh89

#### 1.1.2.2.1.1) Register student in class:

**Description:** This function will register the student to the class.

**Input:** Should need to fill the class registration form

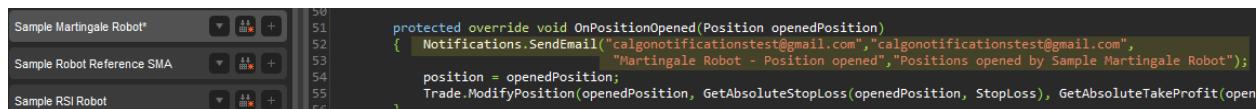
**Output:** Successfully add to the class.

#### 1.1.2.1.a) Activate notifications:

**Description:** This verify the email address and the phone number for notifications

**Input:** Click the button to activate the notifications

**Output:** There will be text message displayed on the device.



```

50
51     protected override void OnPositionOpened(Position openedPosition)
52     {
53         Notifications.SendEmail("calgonotificationstest@gmail.com", "calgonotificationstest@gmail.com",
54             "Martingale Robot - Position opened", "Positions opened by Sample Martingale Robot");
55         position = openedPosition;
56     }

```

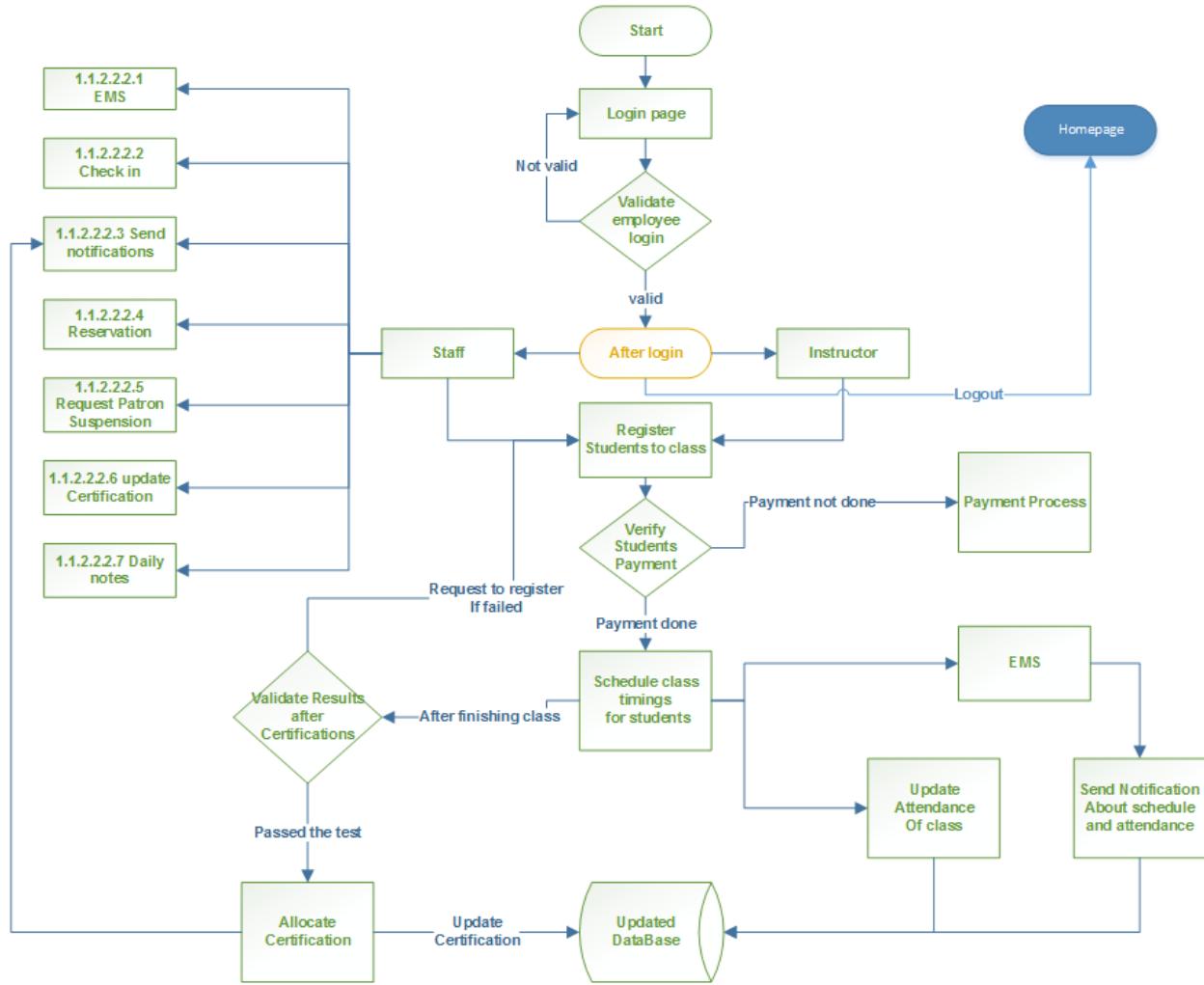
Screenshot 1.10

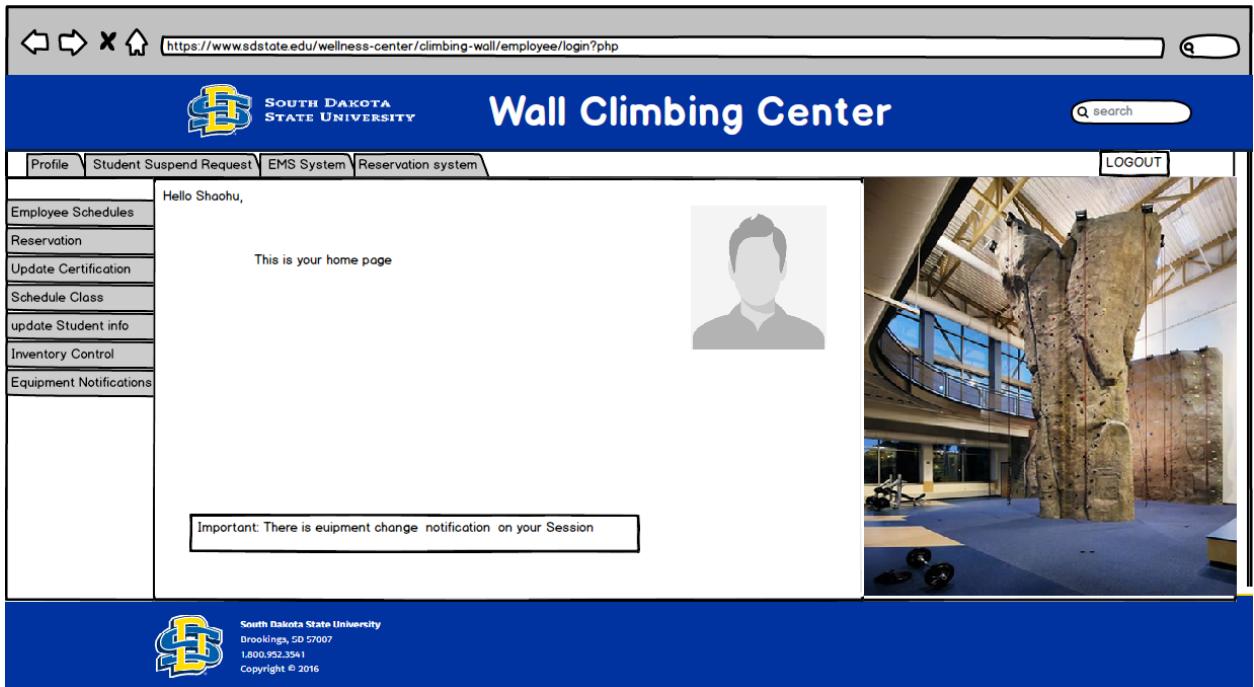
#### 1.1.2.2.1.2.1) Schedule class:

**Description:** This function will schedule the class for the new students who want to climb the wall

**Input:** Fill the registration form and submit it to the staff instructor.

**Output:** You can see your name in the class list of the wall climbing





Screenshot 1.11

#### 1.1.2.2.1.2) EMS system:

**Description:** Display Scheduling System for appointment, Class Timings and the notification remainder and soon.

**Input:** Update the system with a time remainder.

**Output:** Create the schedule and notify the patron employee and the admin for the scheduled information.

#### 1.1.2.2.1.3) Allocate certification:

**Description:** With the allocation of the certification and the update account you can make you climb high levels.

**Input:** Update the certificate (See screenshot 1.11)

**Output:** Eligible for the high grade climbing.

#### 1.1.2.2.2.4) Reservation:

**Description:** Reservation system will be compatible with ems system and the

reservation of class room for the Certification classes and organize some Meetings.

**Input:** The name of the class, instructor and the patron with their requirement date, time and day as input to reserve the room.

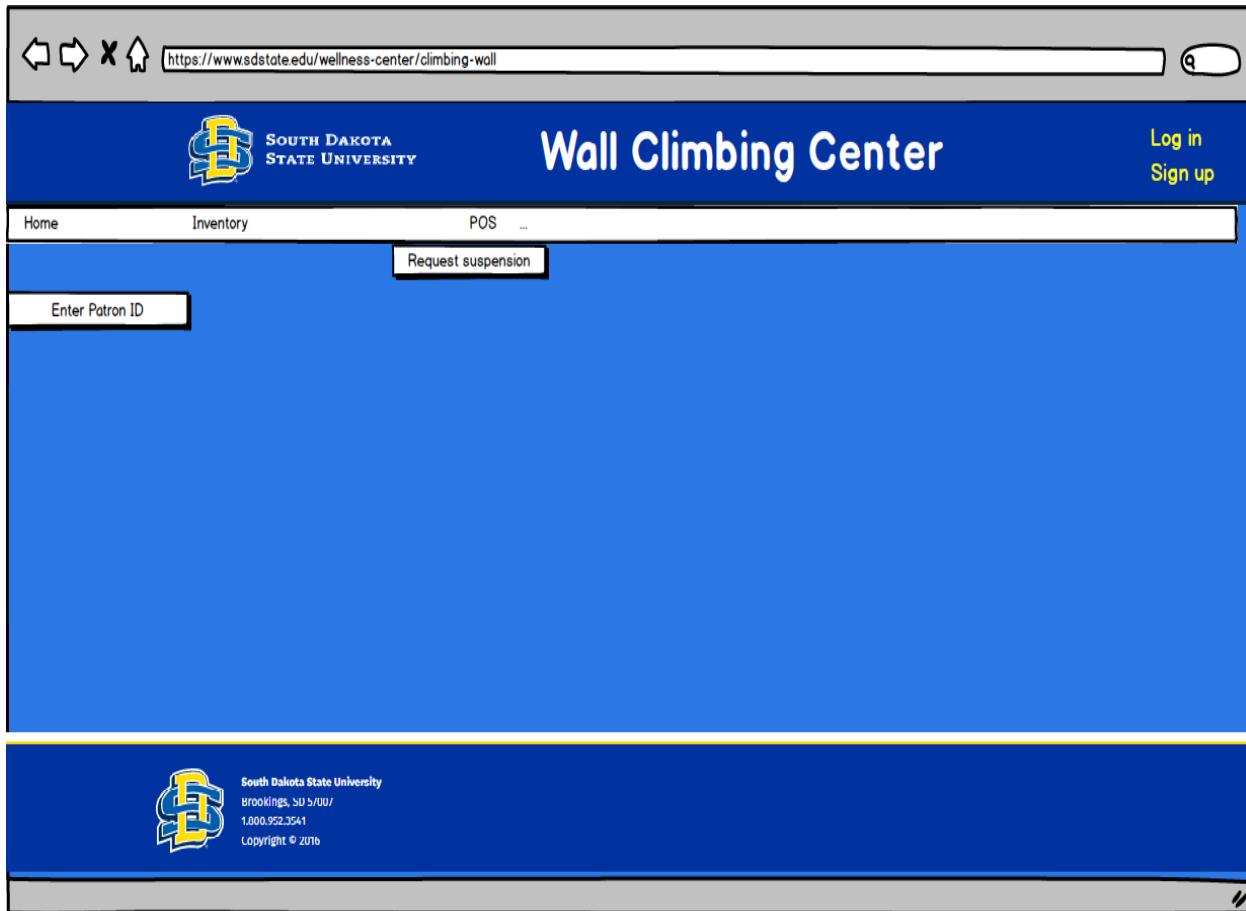
**Output:** Status of the reservation to the staff/ patron.

### 1.1.2.2.2.5) Request patron suspension:

**Description:** Here the employ/staff request the admin for the suspension with reason.

**Input:** the input will be the name of the employee who requested and the patron name, ID, and the reason for request.

**Output:** Email send to admin for the suspension request of patron.



The screenshot shows a web browser window with the URL <https://www.sdsstate.edu/wellness-center/climbing-wall>. The page title is "Wall Climbing Center". The navigation bar includes links for "Home", "Inventory", "POS", and "...". A "Request suspension" button is visible. A dropdown menu titled "Which Policy was broken" is open, listing "Policy One", "Policy Two", and "Policy Three". The footer contains the South Dakota State University logo and copyright information: "South Dakota State University", "Brookings, SD 57001", "1.800.952.3541", and "Copyright © 2016".

The screenshot shows the same web browser window with the same URL and page title. The navigation bar and "Request suspension" button are present. A table is displayed with columns for "Patron ID", "Patron Name", and "Reason of Suspension". Below the table is a "Send a report to Admin" button. The footer is identical to the first screenshot.

#### 1.1.2.2.2.6) Update Certification:

**Description:** After the completion of the class certification the patron will be updated with his certification to his/her profile.

**Input:** Patron certificate.

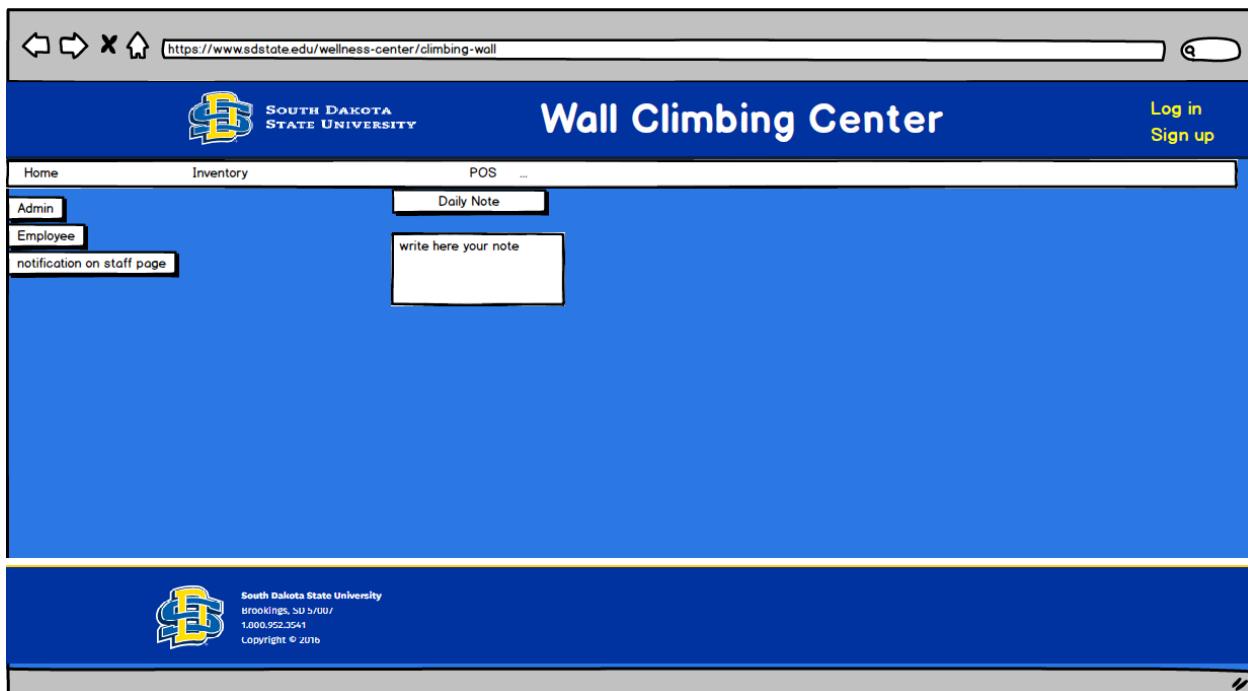
**Output:** Updated database with patron certificate.

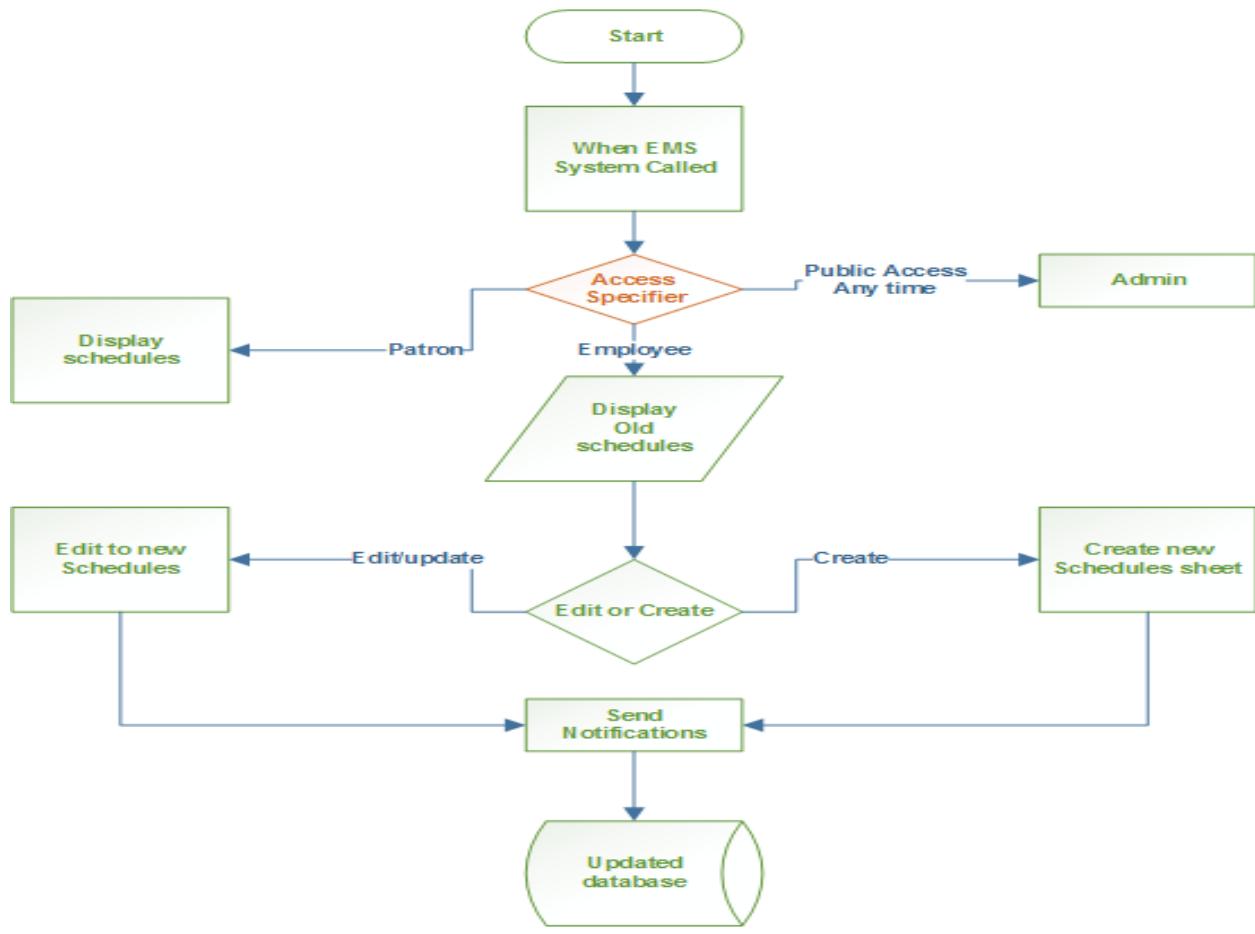
#### 1.1.2.2.2.7) Daily notes:

**Description:** Ability to type any important notes regarding the wall. It will be emailed automatically to staff/admin or read via the program by any staff /admin.

**Input:** Important notes as text written by staff/admin.

**Output:** It will be emailed to staff/admin through their email.



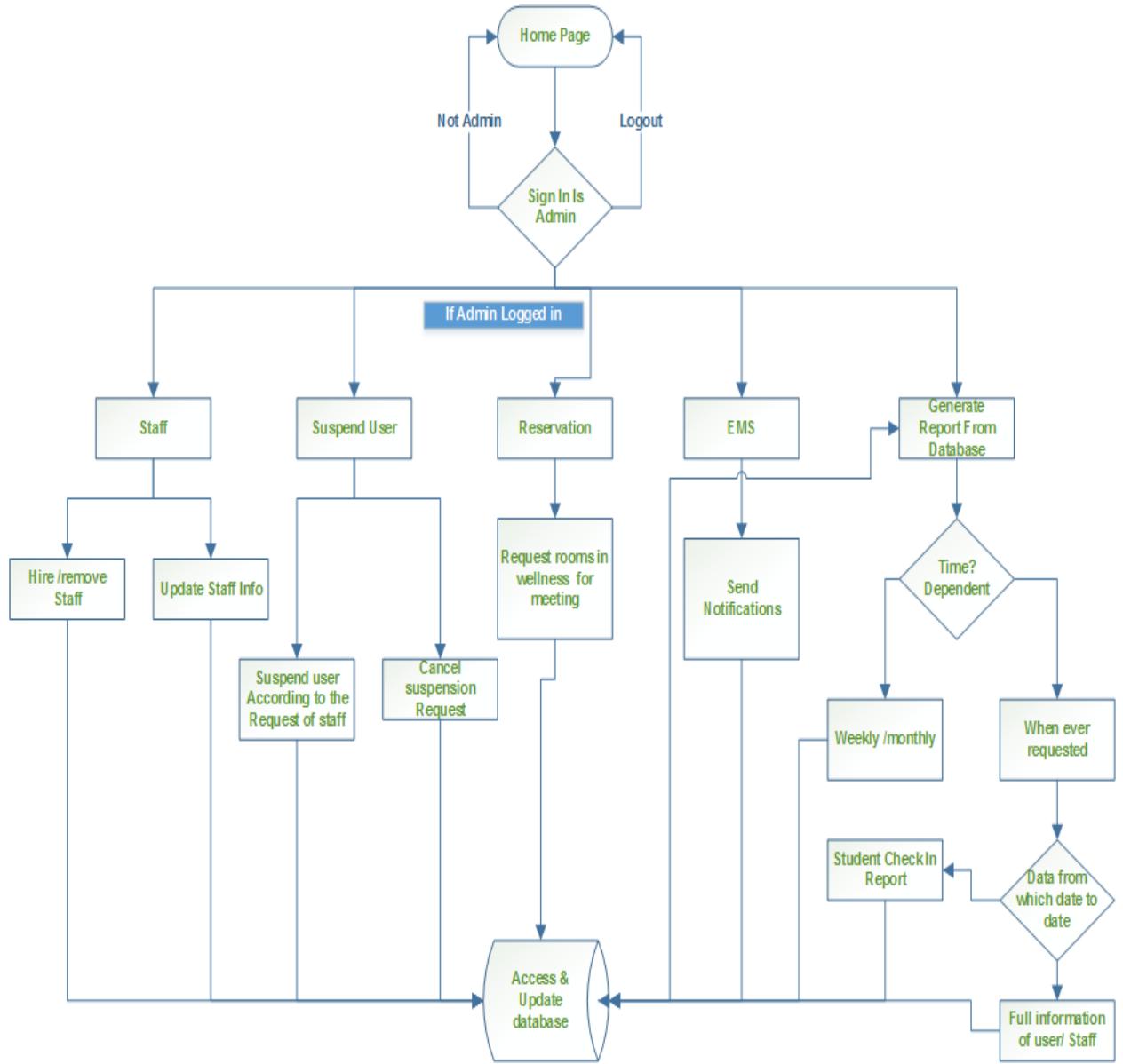


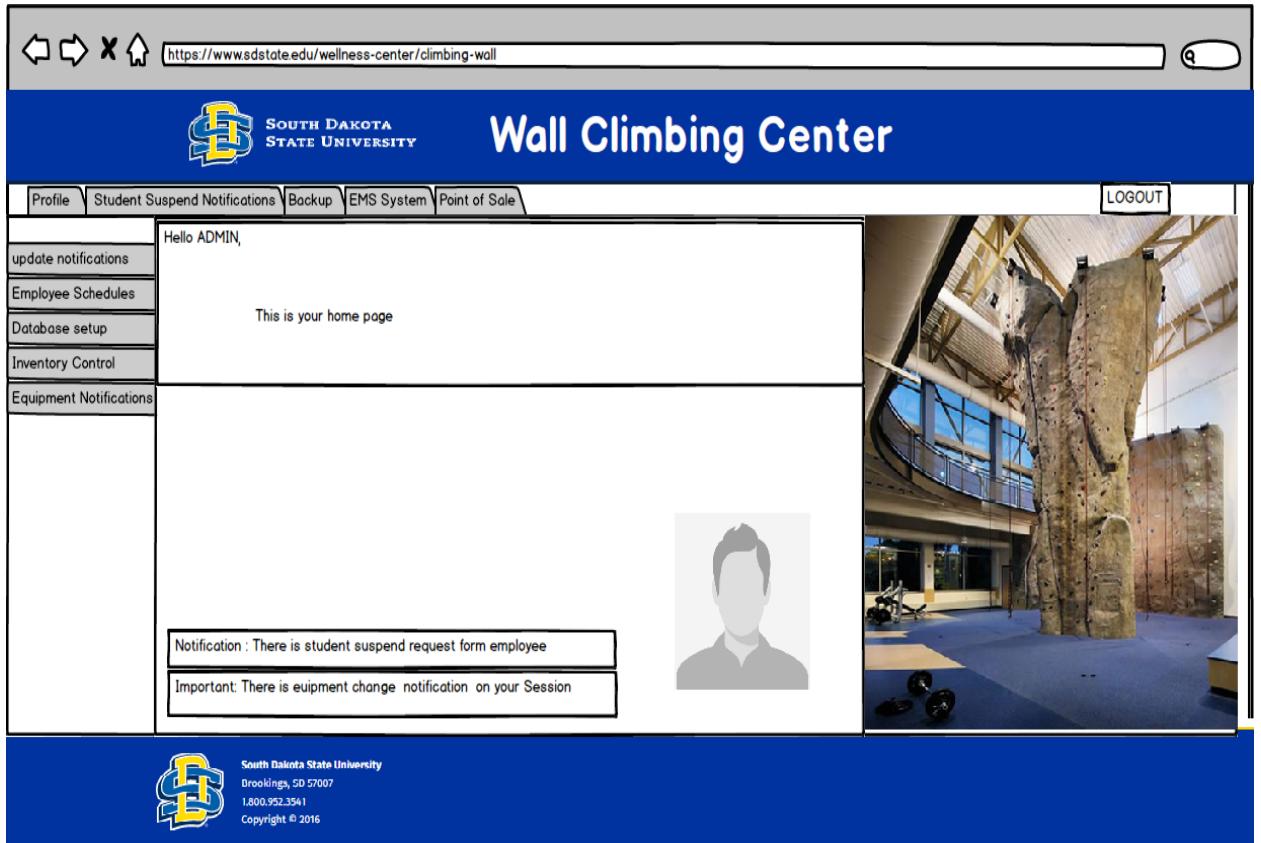
### 1.1.2.3) Administrator:

**Description:** If the Administrator has logged into the system. He can have the Whole Access to the system from registration information to the backup system.

**Input:** Login with the admin username and the password. (see screen 1.2)

**Output:** can get access to the whole system. (see screen 1.12)





**Screen 1.12**

#### 1.1.2.3.1) Hire and remove staff:

**Description:** Can hire or remove staff depending on the admin system manager requirement.

**Input:** Update the details of the staff or remove the details of the staff by admin

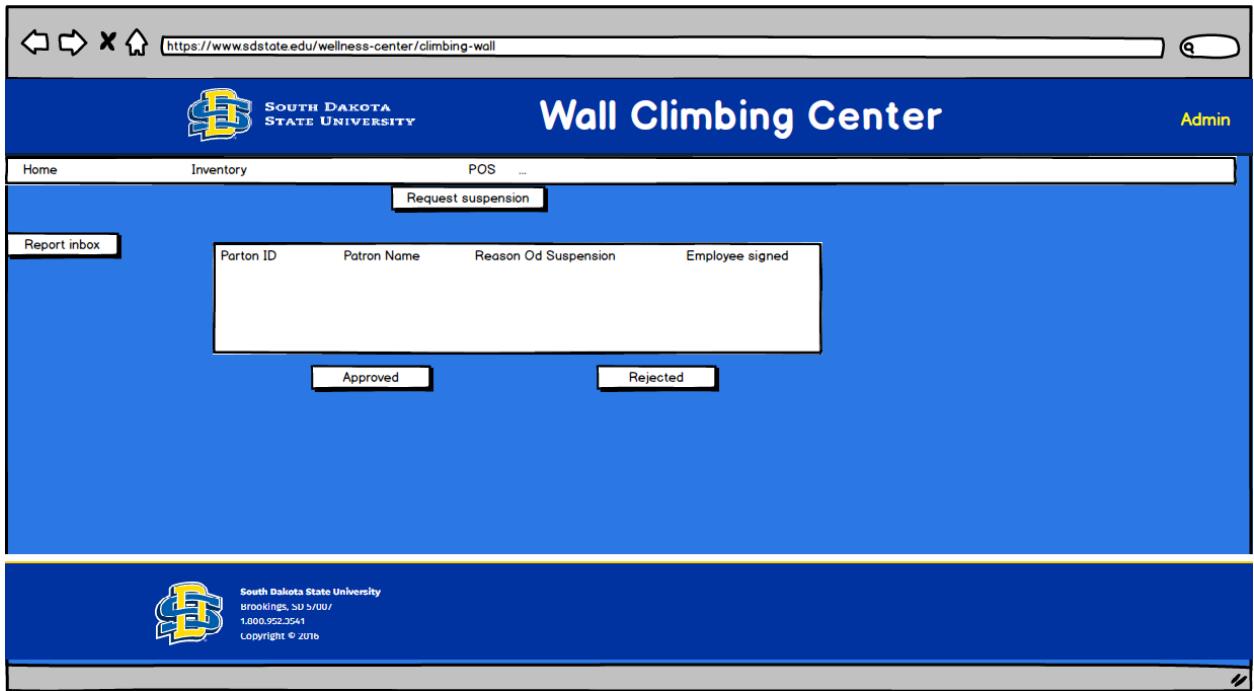
**Output:** update the system.

#### 1.1.2.3.2) Approve user suspension:

**Description:** The patron can be suspended based on the reason of the violation of the rules and the no of times suspended before.

**Input:** Approve suspension. (see screen 1.12)

**Output:** The Patron cannot climb the wall till the suspension is done.



#### **1.1.2.3.3) Reservation:**

**Description:** Reservation system will be compatible with ems system and to book a room for organize some Meetings.

**Input:** The requirement date, time and day as input to reserve the room.

**Output:** Status of reservation.

#### **1.1.2.3.4) EMS:**

**Description:** The admin has the full access to the ems system to modify the schedules and create the new appointment based on his availability and can also create the event based on time of competition.

**Input:** the date, time and the day will be the inputs and the event name sometimes

**Output:** time scheduled event will be the output and it is displayed on the staff / employee profile.

#### **1.1.2.3.5) List Serv:**

**Description:** The admin will have access to the email of the all patron and the staff in order to send the notifications whenever the admin want.

**Input:** All the contact information such as email and the phone number of the patron and the staff.

**Output:** Display list of emails of all patrons and the staff.

#### **1.1.2.3.6) Report generation:**

**Description:** This function will generate reports.

**Input:** select the starting date and the end date.

**Output:** data in excel sheet.

The below screenshots illustrates different reports format.

	A	B	C	D	E	F	G	H	I	J
1	Patron report with total participation									
2	S-NO	First_name	Last_name	Email	phone	expiration date	attended	from_date	End_date	
3	1	Hussain	Otudi	Ot@jacks	605-592	12/16/16	40	10/10/2016	10/16/2016	
4	2	shaoh	zhang	Sh@jacks	605-444	12/17/16	35	10/10/2016	10/16/2016	
5	3	Lee	haney	Lee@jacks	605-232	12/18/16	40	10/10/2016	10/16/2016	
6	4	Shin	sung	Shin@jacks	605-232	12/19/16	32	10/10/2016	10/16/2016	
7	5	Rajo	appala	Rajo@jacks	605-232	12/20/16	27	10/10/2016	10/16/2016	
8	6	Sayun	sahu	Sayun@jacks	605-232	12/21/16	40	10/10/2016	10/16/2016	
9	7	Bader	alzeidi	Bader@jacks	605-232	12/22/16	25	10/10/2016	10/16/2016	
10	8	Ali	selhenia	Ali@jacks	605-232	12/23/16	20	10/10/2016	10/16/2016	
11	9	Min	manki	Min@jacks	605-232	12/24/16	15	10/10/2016	10/16/2016	
12	10	Hamer	george	Hamer@jacks	605-232	12/25/16	40	10/10/2016	10/16/2016	
13										
14										
15										

	A	B	C	D	E	F	G	H	I	J
1	Patron report with unique participation									
2	S-NO	First_name	Last_name	Email	phone	expiration date	attended	from_date	End_date	
3	1	Hussain	Otudi	Ot@jacks	605-592	12/16/16	40	10/10/2016	10/16/2016	
4	2	shaoh	zhang	Sh@jacks	605-444	12/17/16	35	10/10/2016	10/16/2016	
5	3	Lee	haney	Lee@jacks	605-232	12/18/16	40	10/10/2016	10/16/2016	
6	4	Shin	sung	Shin@jacks	605-232	12/19/16	32	10/10/2016	10/16/2016	
7	5	Rajo	appala	Rajo@jacks	605-232	12/20/16	27	10/10/2016	10/16/2016	
8	6	Sayun	sahu	Sayun@jacks	605-232	12/21/16	40	10/10/2016	10/16/2016	
9	7	Bader	alzeidi	Bader@jacks	605-232	12/22/16	25	10/10/2016	10/16/2016	
10	8	Ali	selhenia	Ali@jacks	605-232	12/23/16	20	10/10/2016	10/16/2016	
11	9	Min	manki	Min@jacks	605-232	12/24/16	15	10/10/2016	10/16/2016	
12	10	Hamer	george	Hamer@jacks	605-232	12/25/16	40	10/10/2016	10/16/2016	
13										
14										

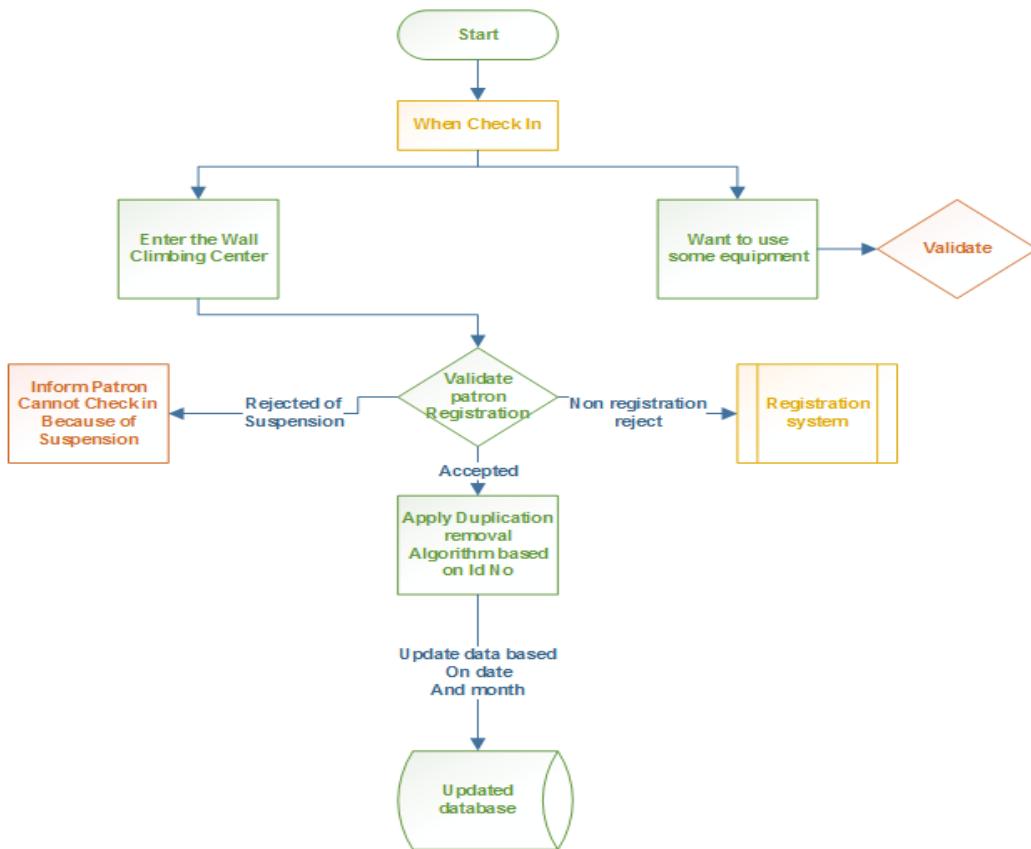
	A	B	C	D	E	F	G	H	I	J
1	Patron report with membership									
2	S-NO	Patron_ID	First_name	Last_name	Email	phone	expiration date	attended	from_date	End_date
3	1	735000	Hussain	Otudi	Ot@jacks	605-592	12/16/16	40	10/10/2016	10/16/2016
4	2	734255	shaoh	zhang	Sh@jacks	605-444	12/17/16	35	10/10/2016	10/16/2016
5	3	735026	Lee	haney	Lee@jacks	605-232	12/18/16	40	10/10/2016	10/16/2016
6	4	735900	Shin	sung	Shin@jacks	605-232	12/19/16	32	10/10/2016	10/16/2016
7	5	739333	Rajo	appala	Rajo@jacks	605-232	12/20/16	27	10/10/2016	10/16/2016
8	6	735343	Sayun	sahu	Sayun@jacks	605-232	12/21/16	40	10/10/2016	10/16/2016
9	7	725353	Bader	alzeidi	Bader@jacks	605-232	12/22/16	25	10/10/2016	10/16/2016
10	8	734666	Ali	selhenia	Ali@jacks	605-232	12/23/16	20	10/10/2016	10/16/2016
11	9	735053	Min	manki	Min@jacks	605-232	12/24/16	15	10/10/2016	10/16/2016
12	10	735949	Hamer	george	Hamer@jacks	605-232	12/25/16	40	10/10/2016	10/16/2016
13										

### 1.1.2.2.2.2) Check in system:

**Description:** Check In System will Validate the Patron when Enter the wall climbing center and also when using some equipment.

**Input:** Enter the ID of the patron. (see screen 1.13)

**Output:** Validate the ID of the patron.





Screenshot 1.13

#### 1.1.2.2.2.2.a) Duplication removal algorithm:

**Description:** This Algorithm will remove the duplicate records of the data.

**Input:** Any Duplicate file record.

**Output:** Remove the duplication.

*Algorithm:*

```

template <class ForwardIterator>
ForwardIterator unique (ForwardIterator first, ForwardIterator last)
{
    if (first==last) return last;

    ForwardIterator result = first;
    while (++first != last)
    {
        if (!(*result == *first)) // or: if (!pred(*result, *first)) for version (2)
            *(++result)=*first;
    }
}

```

```
    }  
}
```

#### 1.1.2.2.5) Request admin for user suspension:

**Description:** In this function the staff or instructor will request the admin for user suspension based upon his violation of the rules in the wall climbing.

**Input:** Request admin for suspension.

**Output:** Admin will review the request.

#### 1.1.1.3.a) Payment encryption method:

**Description:** This encryption will keep the payment details very secret.

**Input:** It is a function that takes the payment details as input.

**Output:** Keep the payment details secure.

#### Encryption:

```
<?xml version='1.0'?>  
<PaymentInfo xmlns='http://example.org/paymentv2'>  
  <Name>John Smith</Name>  
  <CreditCard Limit='5,000' Currency='USD'>  
    <Number>4019 2445 0277 5567</Number>  
    <Issuer>Example Bank</Issuer>  
    <Expiration>04/02</Expiration>  
  </CreditCard>  
</PaymentInfo>
```

#### Card element protection:

```
<?xml version='1.0'?>  
  <PaymentInfo xmlns='http://example.org/paymentv2'>  
    <Name>John Smith</Name>  
    <CreditCard Limit='5,000' Currency='USD'>  
      <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#' Type='http://www.w3.org/2001/04/xmlenc#Content'>  
        <CipherData>  
          <CipherValue>A23B45C56</CipherValue>  
        </CipherData>  
      </EncryptedData>  
    </CreditCard>  
</PaymentInfo>
```

#### 1.1.2.3.3.a) Time of suspension:

**Description:** The time of suspension is based on the number of times the patron was suspended before.

**Input:** It checks the count of the Previous suspensions.

**Output:** Suspension depends on the previous suspensions.

#### **1.1.2.3.3.b) Deactivate account**

**Description:** The previous function time of suspension will deactivate the account until the time of suspension is over.

**Input:** Take time of suspension as input.

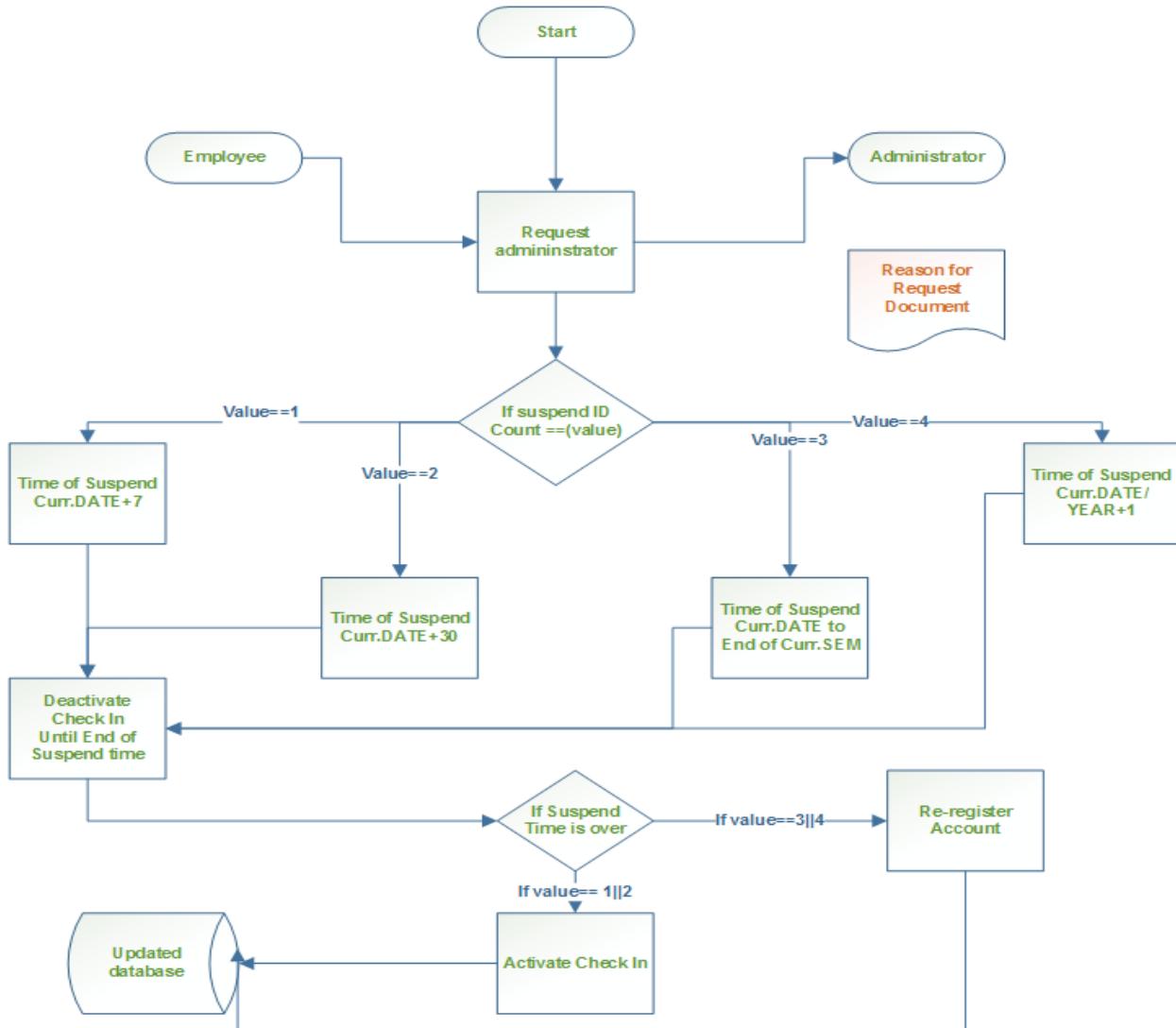
**Output:** Deactivate the user for time of suspension.

#### **1.1.2.3.3.c) Activate check In**

**Description:** This function will activate the check In after the completion of the time of suspension.

**Input:** Deactivate Account

**Output:** Make it active again after suspension time.

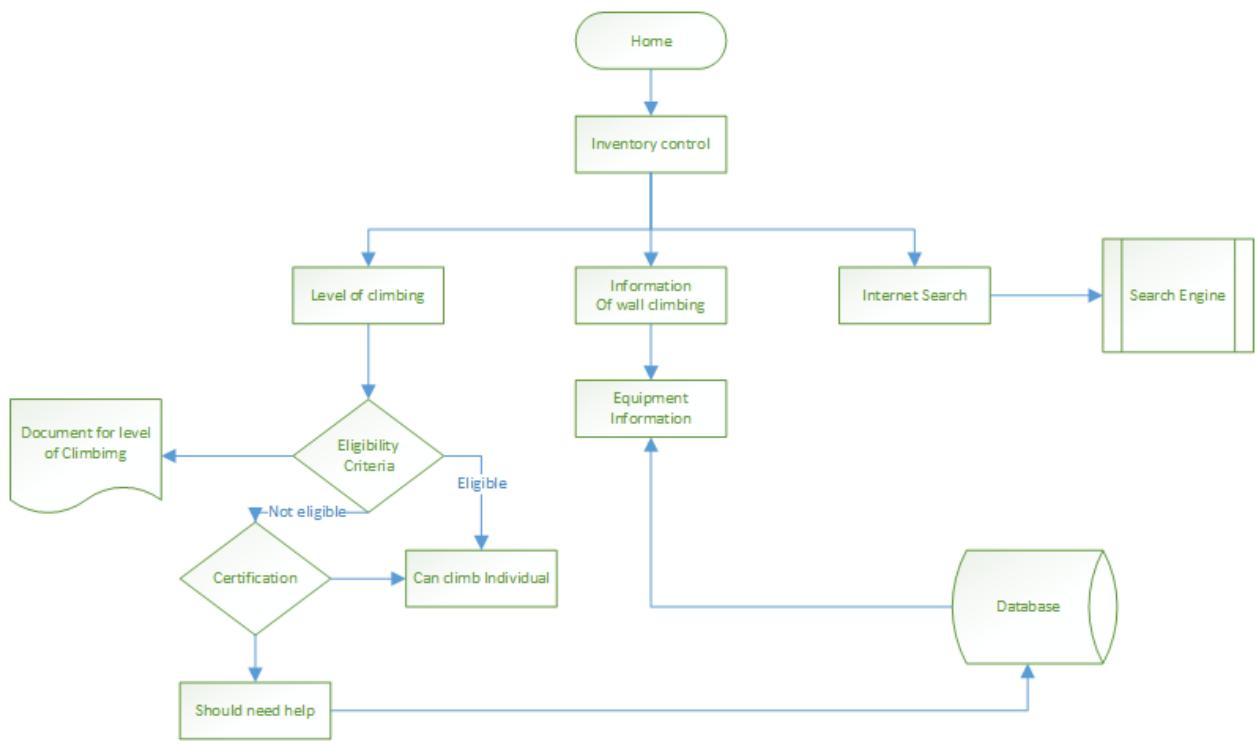


#### 1.1.2.2.2.3) Send notifications:

**Description:** This function will send notifications by verifying the valid email address and the valid phone number.

**Input:** Take valid email or phone number (see screen 1.10)

**Output:** Gives the notification on the device.



### 1.2.2) Level of climbing

**Description:** This describes the level of climbing on the inventory page.

**Input:** No input just the information on the screen.

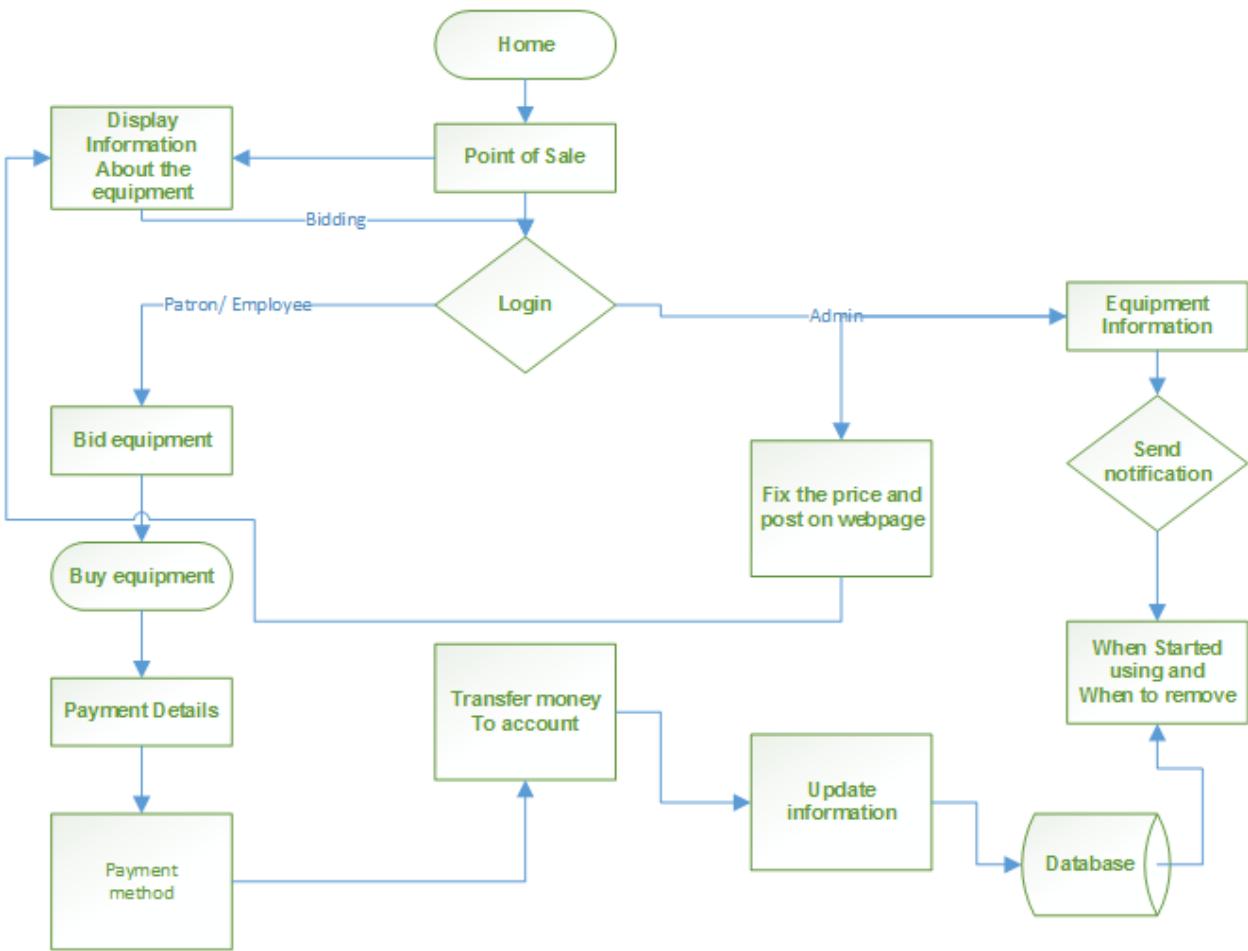
**Output:** Display the output on the screen.

### 1.3.1) Equipment information:

**Description:** Gives the equipment information when it used to bring and fixed and the details of the usage end time.

**Input:** take the date of equipment last date usage.

**Output:** Gives the notification alert on the admin screen.



### 1.3.3) Bid and sell equipment:

**Description:** This function will sell the equipment and let admin know the information of the equipment.

**Input:** Takes the amount of the equipment to sell it

**Output:** After Placing it for bid it will get the amount into Admin Account.

https://www.sdsstate.edu/wellness-center/climbing-wall/Inventory/homepage.php

Equipment Information		Binding Page		
No	Item	Start date	End date	Price
1	Rope Climbing Harness	10/9/2016	11/9/2016	\$99
2	Climbing Harness	10/9/2016	11/9/2016	\$299
3	Chest Harness	10/9/2016	11/9/2016	\$145
4	Climbing Shoes	10/9/2016	11/9/2016	\$199

South Dakota State University  
Brookings, SD 57007  
1.800.952.3541  
Copyright © 2016

POS				
Number	Field	Data Type	Requirement	Example
1	No	Digit	xx	1,2,3,..
2	Items	String	Characters,_ or digit 5<Length<100	shoes
3	Start date	Digit	mm/dd/yyyy	10/12/2016
4	End date	Digit	mm/dd/yyyy	10/11/2016
5	Prise	Double	\$XXX.XX	\$2.333

### 3.2 Non Functional Requirements

- Performance requirements

Table 6. 64-Bit e-Climbing Performance Requirement		
Response Time	Internet	Hardware
system startup: ≤10 sec log in(WBS1.1.2):≤5 sec URL click (WBS1.1; 1.2; 1.3)≤2 sec search inventory item: ≤2 sec	Minimum speed:10Mbs Bandwidth:500 Mbit/s	See Table 2

- Security

Security requirements are:

- The access permissions for system only be approved by administrator
- All system data must be backed up every week and the backup copies stored in a secure location which is not in the same storage disk with the e-Climbing.
- All external communications between the e-Climbing's data server and clients must be encrypted.
- Connection between e-Climbing and the main server has to be encrypted to ensure the safety of the user's information such as credit cards.

- Operational and Environment Requirements

Table 7. 64-Bit e-Climbing Operational and Environment Requirements				
Operating Systems	Database Server	Front-End	Server Side	IDE
Windows 10	MySQL 5.x	JavaScript 1.8 HTML 4.1,5 XML1.0 CSS 3	Java 7.0,8.0	NetBeans 7.2,7.3,7.4,8.0,Java EE bundle

- Backup

The data backup is at **2am** for the specified database as shown in Table 8.

Table 8. Backup plan				
Monday(2am)	Tuesday(2am)	Wednesday(2am)	Thursday(2am)	Friday(2am)
User's information	Inventory data	POS data	Reservation data	History data of Check in

- **Documentation and Training**

e-Climbing system provides online user manual and help. Users can follow the instruction step by step. The user-friendly online page can make it easy for users.

- **Exception Handling**

Table 9. Exception Handling	
Type of Failure	Approach (message to user)
Server failure	Unable to connect to the server
Database failure	Error system is down
Backup failure	Backup error
Internet connection failure	No internet connection
Wrong input	Your input cannot be recognized, please try again

- **Testing Requirements**

Table 10. Testing Requirement and Plan	
Type of Testing	Plan

Installation testing	<ul style="list-style-type: none"> <li>• To test the installation in Windows</li> </ul>
Unit testing	<ul style="list-style-type: none"> <li>• Sign in: verify the user's name and password</li> <li>• Registration: validate the entered user information</li> <li>• Log out: to test if logout successfully</li> <li>• Random ID: to generate a unique ID</li> <li>• Waiver form: input verification, and waiver form generation</li> <li>• Payment: pay type and account information</li> <li>• Class register: to check if it is successful</li> <li>• Certification: to check the level of certification</li> </ul>
Integration testing	<ul style="list-style-type: none"> <li>• Is the filled reservation form stored in database successful?</li> <li>• Does the weekly or monthly report generate completely?</li> <li>• Does the POS system work well with database</li> </ul>
System testing	<ul style="list-style-type: none"> <li>• Test the process of update user information: log in → create/edit information → print → log out</li> <li>• Test the inventory system</li> <li>• Test the check in system: log in, history, certification verification</li> <li>• Test the backup process</li> </ul>

#### 4. LOG OF MEETINGS

<b>9/15 9-11:59PM Thu group meeting</b> 1. RD Discussion 2. Split work	<b>WENS lab</b>	<b>3hrs</b>
<b>9/17 9-11:59AM Sat group meeting</b> Discuss the functionality	<b>WENS lab</b>	<b>3hrs</b>
<b>9/19 9-10:59AM Mon group meeting</b> Introduction complete	<b>WENS lab</b>	<b>2hrs</b>
<b>9/20 9-9:15AM Tue daily scrum meeting</b> <b>Section 2 General description complete</b>	<b>WENS lab</b>	<b>15 mins</b>
<b>9/21 9-9:15AM Wed daily scrum meeting</b> 1. Section 3 complete 2. Polish RD 3. Complete RD draft	<b>WENS lab</b>	<b>15mins</b>
<b>10/2 9pm-12:01am Sun Review</b>	<b>WENS lab</b>	<b>3 hours</b>

**UI design**  
**RD Polish**

**Shaohu:**  
**RD review**

**10/3 5am-12:01pm Mon Review**      **WENS lab**      **8 hours**  
**Polish RD**  
**Make presentation slides**

**10/14 9-9:15 AM Fri Daily Scrum meeting**      **WENS lab**      **15 mins**  
**Attendee: Shaohu, Appala, Hussein**  
**Task:**

- Split work for revision
- Add report format.

**10/16 9-9:15 AM Sun Daily Scrum meeting**      **WENS lab**      **15 mins**  
**Attendee: Shaohu, Appala, Hussein**  
**Task:**  
**Complete the RD**

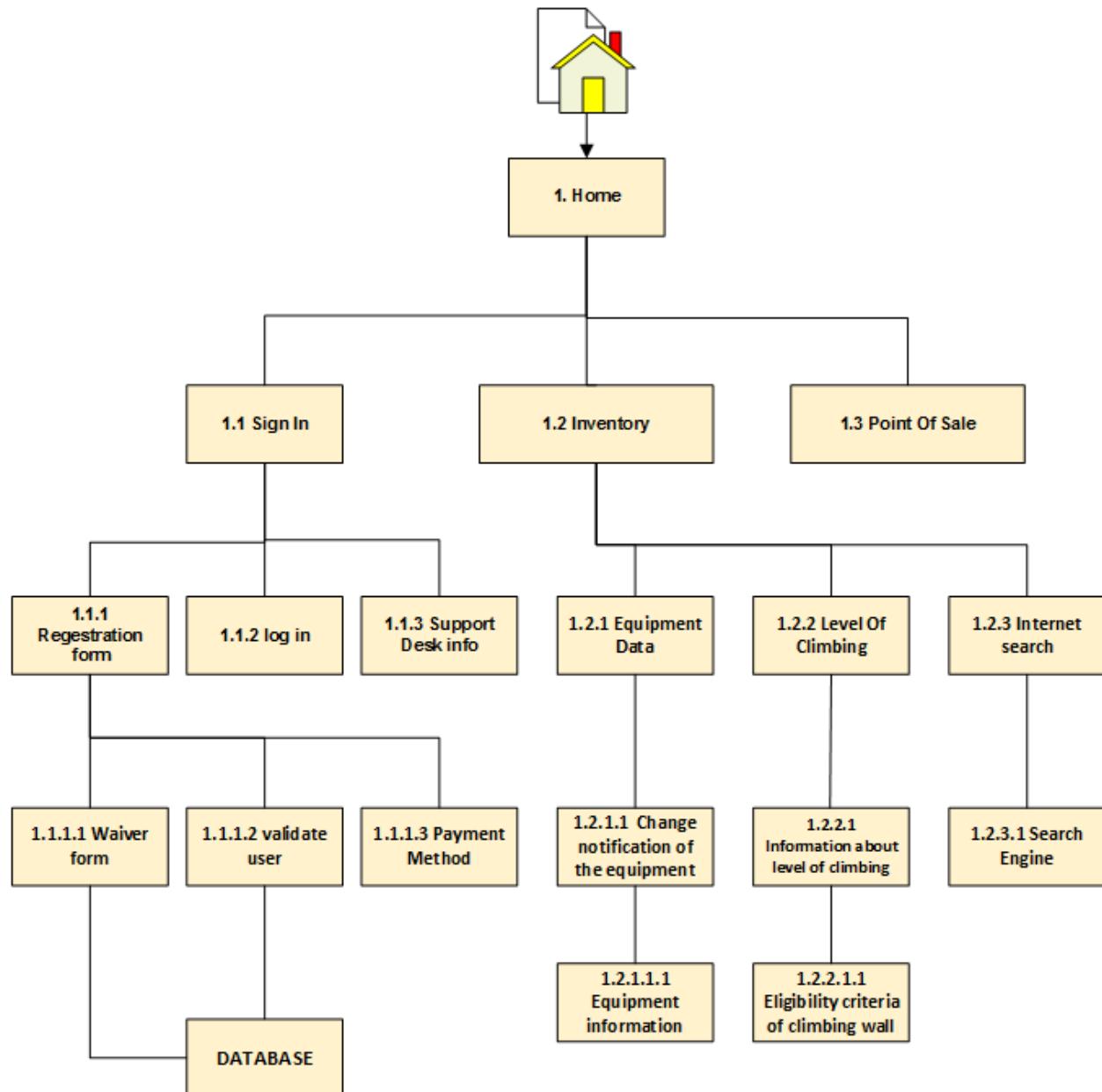
## **5. CHANGE OF CONTROL**

- A Software Version Control(SVC) team should record and control software changes and versions.
- Any changes to this requirements document may be initiated by Mr. Parks and Mr. Shaohu Zhang.
- Any document change should submit to SVC team for 3 days' review.
- Each software change request should be assigned a unique tracking number.
- All files associated with the code must be under version control including software requirements files.
- Code change procedures should encourage frequent code check in.
- Codelines should have policies specific to the reason for their existence.
- Be sure changes are tested.
- Be sure a backup plan exists.
- Inform users.

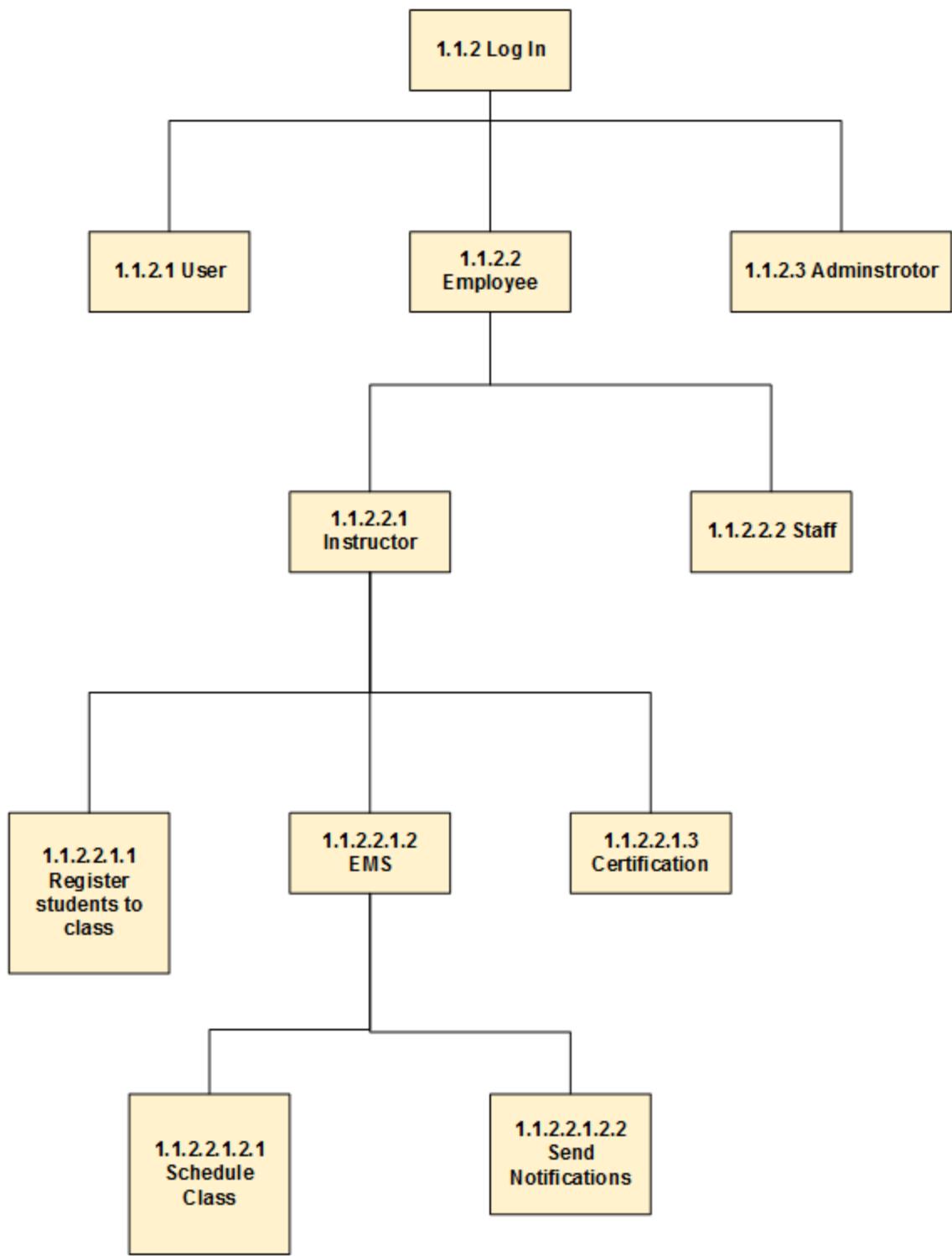
## 6. APPENDIX

### Appendix A. FWBS

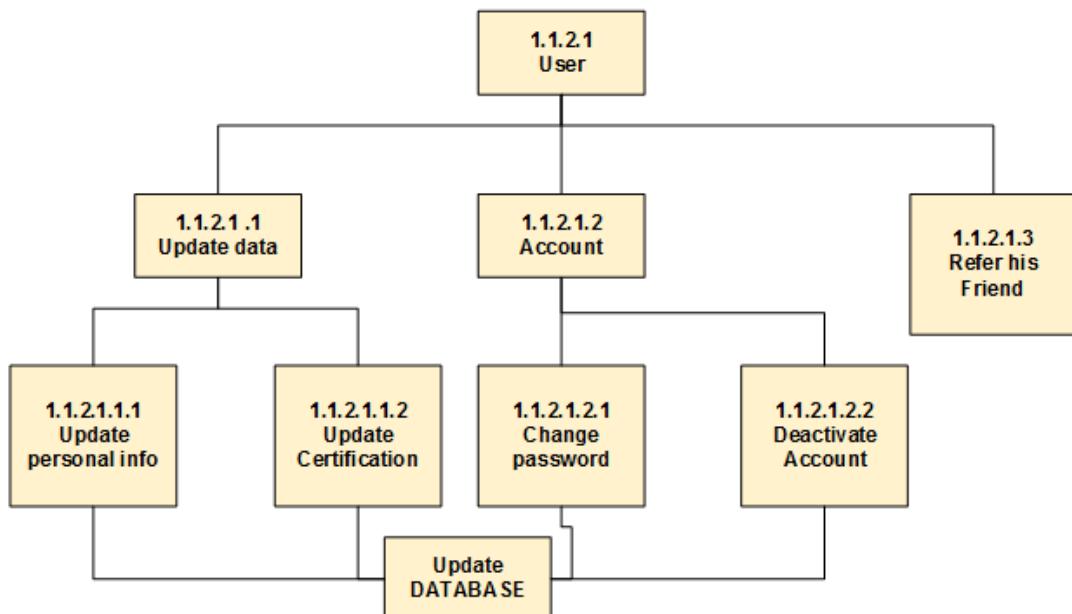
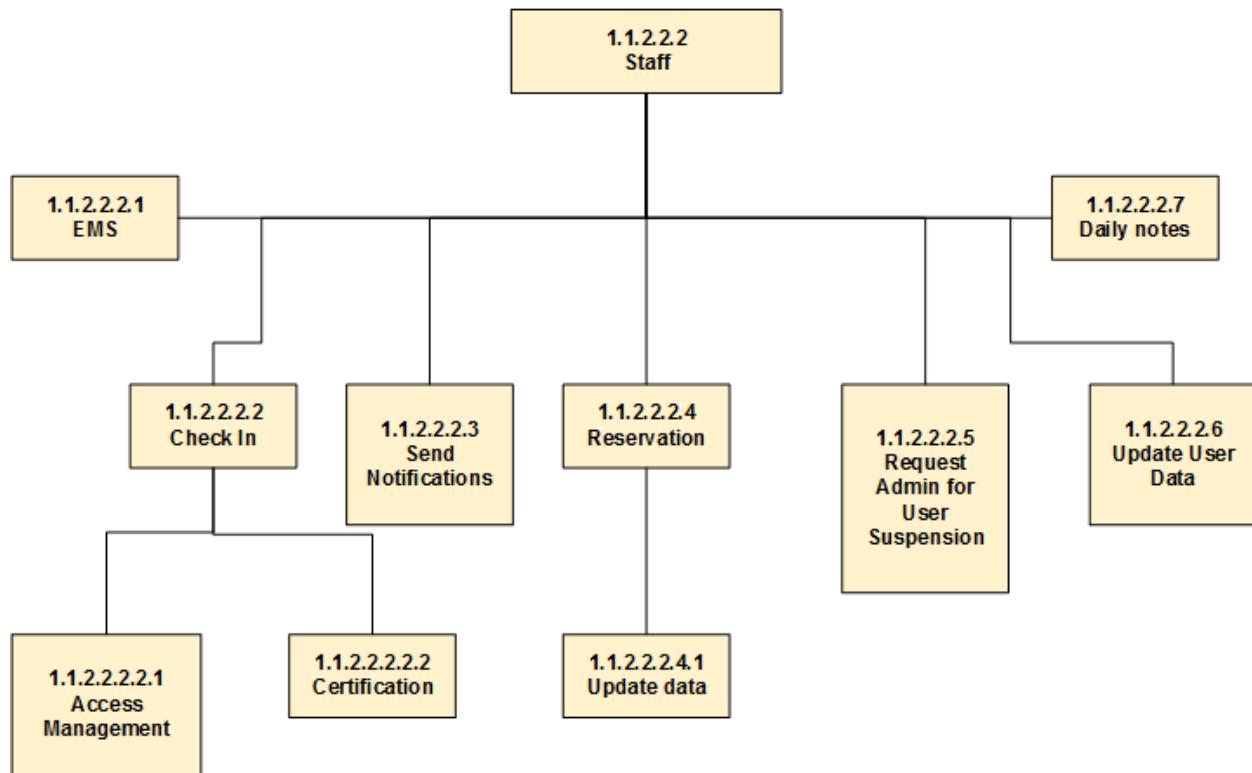
- Main Page:



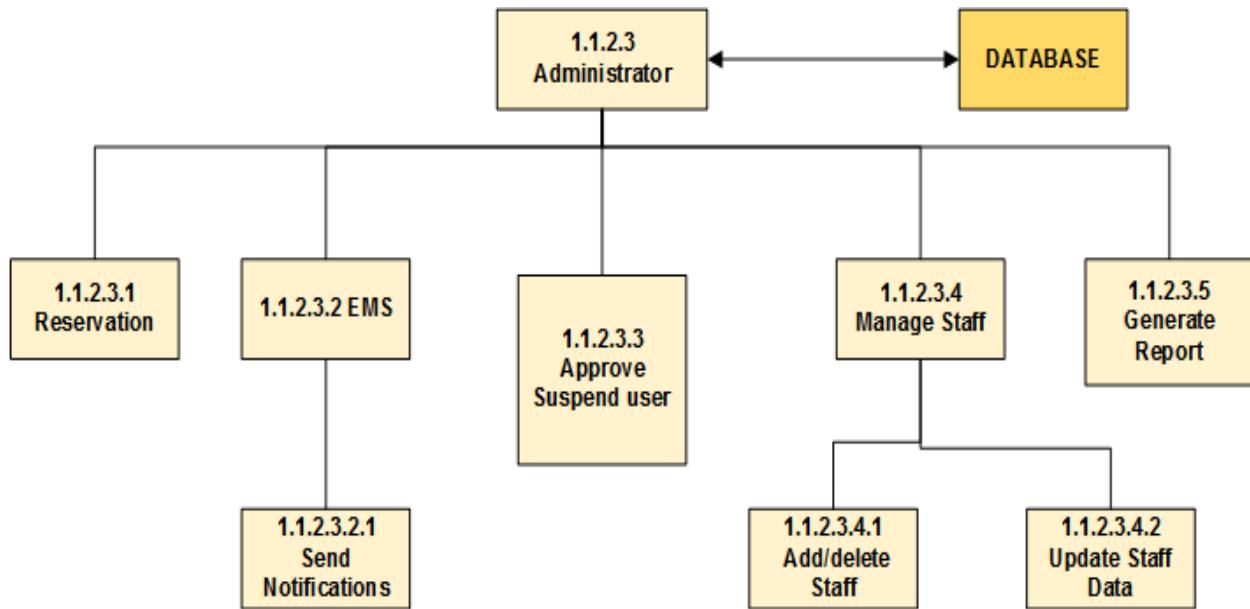
- Login Page:



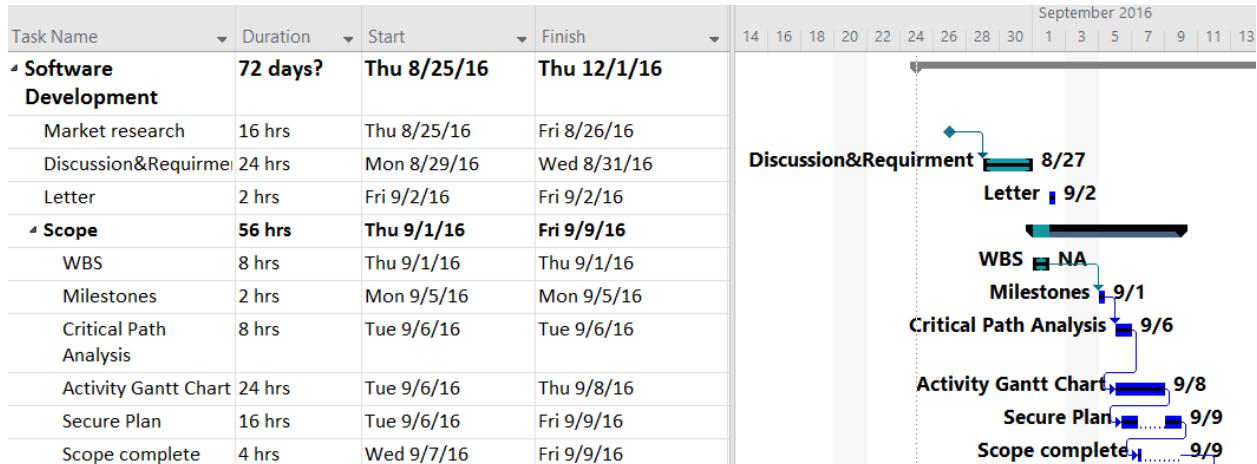
- Staff and User

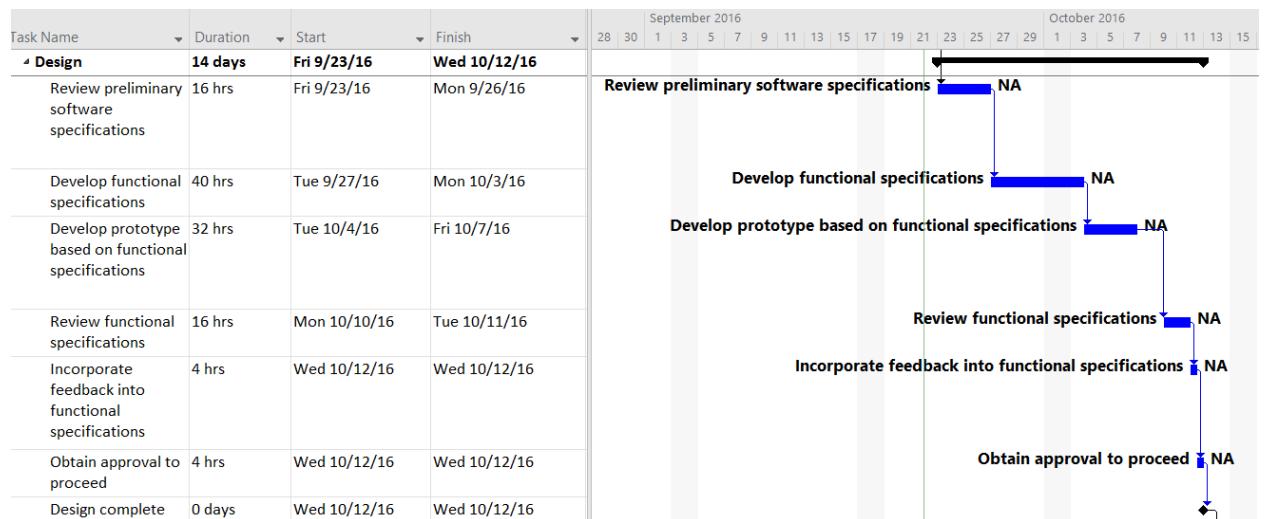
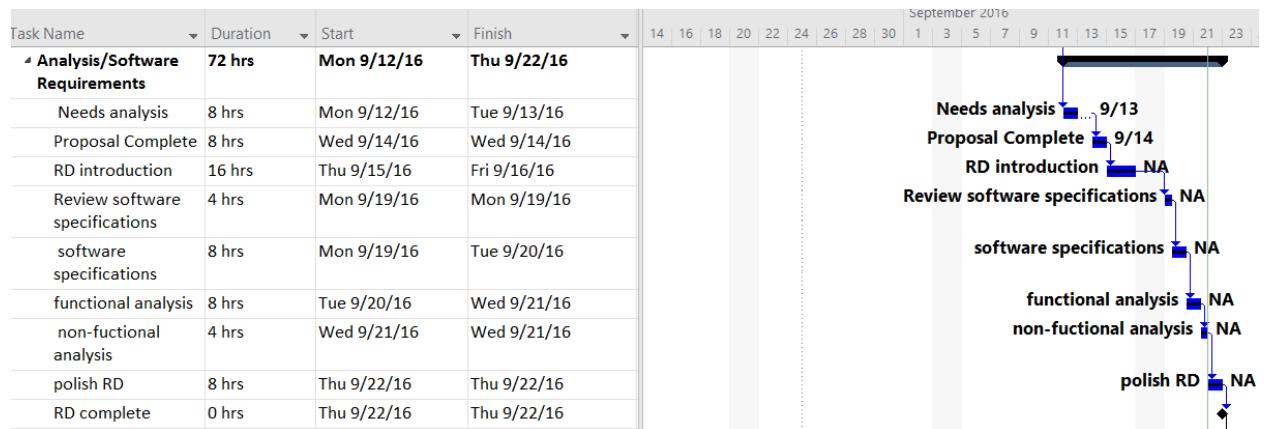


- Admin:



## Appendix B. Project Schedule





Task Name	Duration	Start	Finish	
« Development	20 days	Fri 10/14/16	Thu 11/10/16	
1.Home design	160 hrs	Fri 10/14/16	Thu 11/10/16	
1.1 Sign in design	100 hrs	Mon 10/17/16	Wed 11/2/16	
1.1.1 Registration form	8 hrs	Mon 10/17/16	Mon 10/17/16	
1.1.1.1 Waiver form	8 hrs	Tue 10/18/16	Tue 10/18/16	
1.1.1.2 user validation	4 hrs	Wed 10/19/16	Wed 10/19/16	
1.1.1.3 Payment method	4 hrs	Wed 10/19/16	Wed 10/19/16	
1.1.2 login	8 hrs	Tue 10/18/16	Wed 10/19/16	
1.1.2.1 user	8 hrs	Wed 10/19/16	Wed 10/19/16	
1.1.2.2 employee	4 hrs	Wed 10/19/16	Wed 10/19/16	
1.1.2.3 admin	2 hrs	Wed 10/19/16	Wed 10/19/16	
1.1.2.3.1 reservation	2 hrs	Wed 10/19/16	Wed 10/19/16	
1.1.2.3.2 EMS	5 hrs	Wed 10/19/16	Wed 10/19/16	
1.1.2.3.2.1 email	8 hrs	Wed 10/19/16	Wed 10/19/16	
1.1.2.3.3 suspend user	2 hrs	Thu 10/20/16	Thu 10/20/16	

Task Name	Duration	Start	Finish	
1.1.2.3.4 Manage staff	8 hrs	Fri 10/21/16	Fri 10/21/16	
1.1.2.3.5 Report	8 hrs	Mon 10/24/16	Mon 10/24/16	
1.1.2.3.4.1 new staff	3 hrs	Fri 10/21/16	Fri 10/21/16	
1.1.2.3.4.1 update staff info	1 hr	Fri 10/21/16	Fri 10/21/16	
1.1.2.2.1 instructor	2 hrs	Wed 10/19/16	Wed 10/19/16	
1.1.2.2.2 staff	32 hrs	Thu 10/20/16	Tue 10/25/16	
1.1.2.2.2.1 EMS Schedual	8 hrs	Fri 10/21/16	Fri 10/21/16	
1.1.2.2.2.2 check in	4 hrs	Mon 10/24/16	Mon 10/24/16	
1.1.2.2.2.1.1 class	2 hrs	Thu 10/20/16	Thu 10/20/16	
1.1.2.2.2.1.2 EMS	8 hrs	Fri 10/21/16	Fri 10/21/16	
1.1.2.2.2.1.3 certification	8 hrs	Fri 10/21/16	Fri 10/21/16	
1.1.2.2.2.1.2.1 schedual class	2 hrs	Fri 10/21/16	Fri 10/21/16	
1.1.2.2.2.1.2.2 send notification	8 hrs	Fri 10/21/16	Mon 10/24/16	
1.1.3 support desk	4 hrs	Wed 10/19/16	Wed 10/19/16	
1.2 inventory	16 hrs	Tue 10/25/16	Wed 10/26/16	

Task Name	Duration	Start	Finish	October 2016	November 2016
1.2.1 equipment data	8 hrs	Tue 10/25/16	Tue 10/25/16	1   3   5   7   9   11   13   15   17   19   21   23   25   27   29   31   2   4   6	1.2.1 equipment data ■ NA
1.2.1.1 notification	8 hrs	Tue 10/25/16	Tue 10/25/16		1.2.1.1 notification ■ 10/25
1.2.1.1.1 equipment information	8 hrs	Fri 10/14/16	Fri 10/14/16		Equipment information ■ NA
1.2.2 level of climbing	8 hrs	Fri 10/28/16	Fri 10/28/16		1.2.2 level of climbing ■ 10/28
1.2.2.1 level information	8 hrs	Fri 10/28/16	Fri 10/28/16		1.2.2.1 level information ■ 10/31
1.2..2.1.1 eligibility	4 hrs	Mon 10/31/16	Mon 10/31/16		1.2..2.1.1 eligibility ■ 10/31
1.2.3 internet search	2 hrs	Mon 10/31/16	Mon 10/31/16		1.2.3 internet search ■ 10/31
1.2.3.1 search engine	1 hr	Mon 10/31/16	Mon 10/31/16		1.2.3.1 search engine ■ 10/31
1.3 POS	32 hrs	Thu 10/27/16	Tue 11/1/16		1.3 POS ■ 10/27
1.3.1 equipment	8 hrs	Thu 10/27/16	Thu 10/27/16		1.3.1 equipment ■ NA
1.3.1.1 equipment form	2 hrs	Fri 10/28/16	Fri 10/28/16		1.3.1.1 equipment form ■ NA
1.3.2 sell items	4 hrs	Fri 10/28/16	Fri 10/28/16		1.3.2 sell items ■ 10/28
1.3.2.1 online equipment	4 hrs	Fri 10/28/16	Fri 10/28/16		1.3.2.1 online equipment ■ NA
Task Name	Duration	Start	Finish	October 2016	November 2016
1.3.2.1.1 search design	1 hr	Sat 10/29/16	Sat 10/29/16	5   7   9   11   13   15   17   19   21   23   25   27   29   31   2   4   6   8   10   12   14   16	1.3.2.1.1 search design ■ NA
1.3.3 budget management	7 hrs	Tue 11/1/16	Tue 11/1/16		1.3.3 budget management ■ NA
1.3.3.1 equipment storage	3 hrs	Tue 11/1/16	Tue 11/1/16		1.3.3.1 equipment storage ■ 11/1
1.3.3.2 profit	3 hrs	Tue 11/1/16	Tue 11/1/16		1.3.3.2 profit ■ 11/1
Testing	2 days	Wed 11/9/16	Thu 11/10/16		
Develop unit test plans using product specifications	4 hrs	Wed 11/9/16	Wed 11/9/16		Develop unit test plans using product specifications ■ 11/9
Develop integration test plans using product specifications	4 hrs	Wed 11/9/16	Wed 11/9/16		Develop integration test plans using product specifications ■ 11/9
Integration Testing	5 days	Wed 11/9/16	Tue 11/15/16		
Test module integration	40 hrs	Wed 11/9/16	Tue 11/15/16		Test module integration ■ NA
Identify anomalies to specifications	16 hrs	Wed 11/9/16	Thu 11/10/16		Identify anomalies to specifications ■ NA
Modify code	24 hrs	Wed 11/9/16	Fri 11/11/16		Modify code ■ NA
Task Name	Duration	Start	Finish	2016	December 2016
Re-test modified code	16 hrs	Wed 11/9/16	Thu 11/10/16	5   7   9   11   13   15   17   19   21   23   25   27   29   31   2   4   6   8   10   12   14   16   18   20   22   24   26   28   30   2   4   6	Re-test modified code ■ NA
Integration testing complete	4 hrs	Wed 11/9/16	Wed 11/9/16		Integration testing complete ■ NA
Documentation	72 days	Thu 8/25/16	Thu 12/1/16		
Proposal	17 days	Thu 8/25/16	Thu 9/15/16		
Requirement document	6 days?	Thu 9/15/16	Thu 9/22/16		
Design specification	11 days?	Thu 9/22/16	Thu 10/6/16		■ 10/6
Acceptance test plan	3.5 wks	Tue 10/18/16	Thu 11/10/16		Acceptance test plan ■ 11/10
System test Plan	5 days	Fri 11/11/16	Thu 11/17/16		System test Plan ■ 11/17
User guide	10 days	Fri 11/18/16	Thu 12/1/16		User guide ■ 12/1

## **Appendix C. Terms of Acceptance**

Nester Software shall use commercially reasonable efforts to deliver to end user, no later than 7 days after the Effective Date OR as soon as commercially practicable. For each Deliverable under this Agreement, End user shall have a 14 day "Acceptance Period" beginning on the Delivery Date. During the Acceptance Period, Customer may cancel the license by giving written notice to Vendor and returning the Deliverable in a commercially reasonable manner. Unless such cancellation notice is given, the license will be deemed accepted by Customer at the end of the Acceptance Period

## **Appendix D. Terms and Condition**

- **General**

This Agreement, upon acceptance by you (the "End User"), which acceptance shall be indicated by signature or other recorded End User confirmation, forms a legally binding agreement between the End User and Nester

This Agreement shall take precedence over any terms and conditions sought to be relied upon by the End User in respect of the Online Service. Notwithstanding any language on any correspondence received from the End User to the contrary, this Agreement shall take precedence over any such correspondence. Such correspondence shall be accepted by Nester for administrative purposes only and shall not modify or amend this Agreement. All terms and conditions on any correspondence originating from the End User shall be null, void and without legal effect.

- **Grant of License to End User**

a. License grants are subject to the End User's obligation to pay and continue paying the Subscription Charges and the End User's compliance with this Agreement and any additional product use terms associated with this Agreement.

b. Limitations on use. Licensed software is licensed to the End User, not sold. The End User has no right to:

- i. reverse engineer, decompile, or disassemble any licensed software, except where applicable law permits it despite this limitation;
- ii. rent, lease, lend, resell, or host to or for third parties any licensed software, except as may be expressly permitted for a given licensed software in the product use terms.

Each party represents and warrants that on this date they are duly authorized to bind their respective principals by their signatures below.

**Customer:**

---

**(Signature)**

---

**(Typed or Printed Name)**

Title: \_\_\_\_\_

Date: \_\_\_\_\_

Developer:

---

**(Signature)**

**(Typed or Printed Name)**

Title: \_\_\_\_\_

Date: \_\_\_\_\_

# Design Document

## e-Climbing System

**Prepared For:**  
The Wellness Center Wall Climbing  
of  
South Dakota State University



**Version 2.0**

**Prepared by:**  
**Nester Software Inc.**

**1400 8th St. Nester Center  
Brookings, SD 57006  
10/2016**

**Approved for public release; distribution is unlimited**

### Authoring & Approval

Name	Position & Title	Responsibility	Date
Shaohu Zhang	CEO Nester Software Inc.	Author	10/24/2016
Appala Chekuri	Software project manager Nester Software Inc.	Author	10/24/2016

Hussein Otudi	Systems Engineer Manager Nester Software Inc.	Author	10/24/2016
Justin Park	South Dakota State University	Approval	10/24/2016

## Reviewer

Name	Position & Title	Responsibility	Date
Shaohu Zhang	CEO Nester Software Inc.	Reviewer	10/24/2016
Hussein Otudi	Systems Engineer Manager Nester Software Inc.	Reviewer	10/24/2016

## Revision History

Date	Description	Revision	Editor
10/3/2016	Created the document	0	Hussein, Shaohu
10/10/2016	Added introduction	0.1	Shaohu
10/11/2016	Added design priority table	0.2	Shaohu
10/12/2016	Added DFD	0.3	Hussein, Appala
10/12/2016	Added Design Overview	0.4	Shaohu
10/13/2016	Added Design Priority Section Shaohu <ul style="list-style-type: none"> <li>• Error Handling</li> <li>• Design Constraints</li> <li>• Development and Execution Environment</li> </ul> Bader <ul style="list-style-type: none"> <li>• Programming tools</li> </ul>	0.5	Shaohu
10/14/2016	Added revised DFD	0.6	Hussein, Appala Shaohu
10/15/2016	Added revised report format	0.7	Appala, Hussein
10/16/2016	Added median design level	0.8	Hussein, Appala Shaohu

10/16/2016	Formatting, Reviewer	0.9	Shaohu
10/17/2016	Reviewer	1.0	Hussein
10/23/2016	Revised priority table	1.1	Shaohu
10/24/2016	Revised DFD	1.2	Appala, Hussein
10/24/2016	Review: Format Generated TOC	2.0	Shaohu

**This deliverable has two hard copies. One is for Dr. Shin from South Dakota State University, and another one is submitted to Mr. Park for approval. Distribution is unlimited for public release.**

**SOFTWARE DEVELOPMENT DESIGN DOCUMENTATION**

**FOR**

**E-CLIMBING SYSTEM**

**Prepared by:**  
**Nester Software Inc.**  
1800 8th St. Nester Center  
Brookings, SD 57006  
10/2016

**Team Information**

*Appala Chekuri*

---

**Software Project Manager** **Signature**

*Hussein Otudi*

---

**Systems Engineer Manager:** **Signature**

*Shaohu Zhang*

---

**Chief Executive Officer** **Signature**

# CHAPTER 3. DESIGN DOCUMENTATION

## 1. INTRODUCTION

### 1.1 Purpose

This document sets forth a description of the system design and implementation details for the “e-Climbing” software application in accordance with the requirements listed in the Software Requirement Documentation. This document describes the detailed software features and overall architecture for the system to be built. The Software Design Document(SDD) should help in understanding the overall structure of the e-climbing framework and architecture and will be helpful to current or future developers , testers or users who may work and/or extend it.

### 1.2 Major Problems

The current system in Wall Climbing Center (WCC) has the following issues and challenges:

- **Patron/employee accessibility**

There is no online access system for customer and employees. The customers have to call the program manager for reservation. The available time slot is not accessible for public.

Program manager is the only person for dealing with reservation and registration. The class schedule and flyer only can be grabbed in WCC.

- **Lack of patron evolvement**

The current system only has employee and admin domain. Patron is unable to register or make reservation online.

- **Automatic report generation**

The center uses Excel and EMS for the main tool for daily operation and management. It is very inefficient. For example, the staff has to check register form or visitor log one by one to get weekly or monthly report. It is really time-consuming.

### 1.3 Project Goals

Our goal is not only to solve these issues as much as possible, but also will develop an intelligent and efficient wall climbing management system. These goals include:

- **Patron/employee accessibility**

The system will include more accessibility for Patron and employee.

- **Online reservation**

Users can make reservation on internet by computer, tablet or smartphone.

- **Online waiver**

The system should have ability to accept online waiver. The complete waiver form will be automatically saved in department driver.

- **Automatic report generation**

The system will have fully ability to generate a wide array of report based upon facility access such as user information, participation, new users and so on.

- **Suspension management**

Employees have ability to request suspension of Climbing Wall privileges for individuals that broke policy with an included incident report to describe why the request is being submitted.

Admin have ability to suspend access to the Climbing Wall based upon the incident report generated by the staff member.

System will deny access until the date that the suspension is over.

System will automatically increase suspension by a certain percentage if the individual is a repeat offender.

- **Certification management**

The system tracks whether or not the individual has a current belay or lead climb certification with the climbing gym. The system gives warning of retest if the individual has not checked in at the wall for a period of over three months.

## 1.4 Definitions, Acronyms, and Abbreviations

Table 1. Definitions, Acronyms, and Abbreviations

Terms	Definition, Acronym, Abbreviation
Architecture	The structure or structures of the system. They comprise software components, the externally visible properties of those components, and the relationships among them.
Class	It is a construct that is used to create instances of itself.
DFD	Design Flow Diagram
I/O	Input Output
Framework	Represents a collection of conceptual and technology mechanisms that provide a set of services and guidelines for applying a problem to particular domain.
GPU	Graphics Processing Unit
WBS	Work Breakdown Structure
SDD	Software Design Document
EMS	Email Management System

DOB	Date of Birth
DB	Database
UML	Unified Modeling Language

## 1.5 Reference

[1].Software Design Document

<https://www.cs.drexel.edu/~dpn52/Therawii/design.pdf>

[2]. Fill Out form

<http://www.trigent.com/blog/java-application-development-automatic-form-filling-in-pdf-using-java-codes/>

[3] waiver accept reject form

<https://coderanch.com/t/492680/java/Simple-Accept-Reject-Dialog-refuses>

[4] Add/delete Employee from the database

<http://codereview.stackexchange.com/questions/80371/creating-and-deleting-employees-in-a-database>

[5] payment method

<https://www.simplify.com/commerce/docs/tutorial/index>

[6] Report generation

<http://www.codejava.net/coding/how-to-write-excel-files-in-java-using-apache-poi>

[7] EMS Scheduling

<https://msdn.microsoft.com/en-us/library/gg334289.aspx>

## 1.6 Overview

This document presents detailed information relating to the design for the e-Climbing software as shown in the following:

- Section 1 presents a brief overview of e-Climbing and the issues driving its desire for development of the application.
- Section 2 lays out the minimum and recommended hardware and software required for a proper configuration of the e-Climbing system.
- Section 3 considers high-level issues relating to e-Climbing design and development.

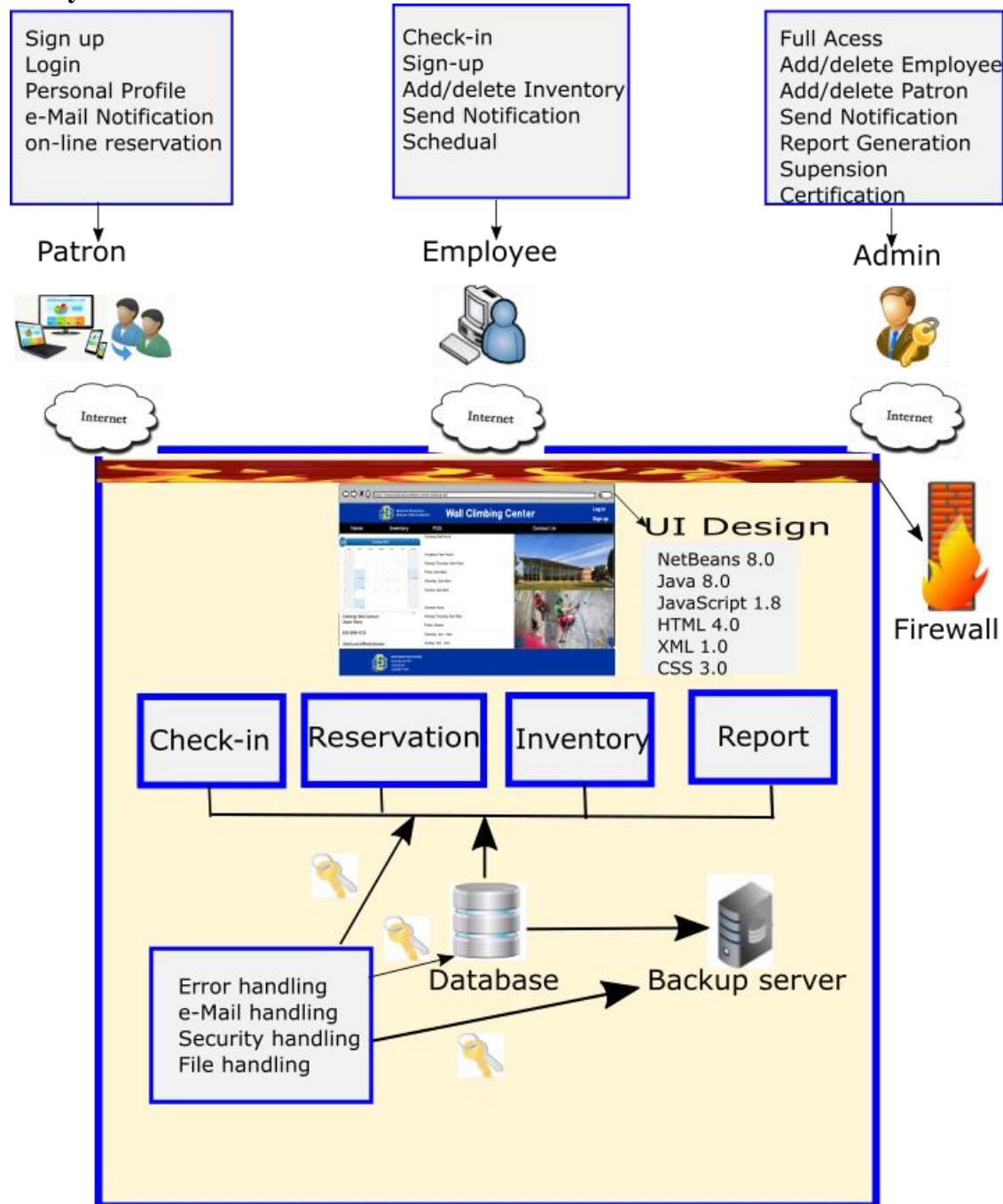
- Sections 4 and 5 present the established design. Section 4 presents the high-level overall design concept for each e-Climbing component, while Section 5 considers each component's design in detail.

Additional information relating to e-Climbing design/implementation is presented in various Appendices.

## 2. DESIGN OVERVIEW

This section describe the hardware and system operational platforms intended for e-Climbing.

### 2.1 System Architecture



## 2.2 System Operation

Table 2 shows the minimum and recommended hardware configuration for the server(s) to run e-Climbing.

Table 2. Minimum/Recommended e-Climbing Server Hardware Configuration					
Hardware Component	Processors	Disk Space	RAM	Graphics	Network Card
Minimum	Any Intel or AMD x86-64 processor	1 GB for e-Climbing only, 2–4 GB for a typical installation	4 GB is required	No specific graphics card is required.  Hardware accelerated graphics card supporting OpenGL 3.3 with 1GB GPU memory	0.5 Gbit/s Ethernet
Recommend	Intel or AMD x86-64 processor with 2.5 Ghz CPU	500 GB or greater	8 GB or greater	Hardware accelerated graphics card supporting OpenGL 3.3 with 2 GB GPU memory is recommended.	1 Gbit/s Ethernet

Table 3 shows the recommended software configuration for the server(s) to run e-Climbing.

Table 3. Recommended e-Climbing Server Software Configuration		
Software Component	Web Server	Database Server
Operating System	Windows 10	Windows 10
Antivirus	Norton Antivirus 2015 or greater, Symantec Endpoint Protection 11	Norton Antivirus 2015 or greater, Symantec Endpoint Protection 11
Web	Apache 2.2.11 or greater	NA

Database	NA	MySQL 5.0 or later
Java	Java 7.0 or later	Java 7.0 or later

### 3. DESIGN PRIORITY

The design priority is to ensure that the system design meets the aspects of the customer, user and the overall business market.

#### 3.1 Design priority table

Priority is defined as the characteristic. We set a priority range from 1-10, where 1 is the lowest and 10 is the highest. Table 4 gives the design priority table of system design. Design 1 is the priority score for the web-based application with internet and remote database backup. Design 2 is the score for desktop application with remote database backup. Design 3 is for the desktop application without remote database backup.

Table 4. Design priority table				
Design Items	Weight(1-10)	Design 1	Design 2	Design 3
Reliability	10	6	10	9
Performance	9	8	8	9
Execution speed	8	10	6	8
Integrity	9	7	6	8
Back up	8	6	6	7
User friendliness	6	7	8	8
Error Handling	6	8	7	8
Modifiability	6	5	8	7
Testability	6	5	6	7
System cost	5	7	6	5
Total	---	508	526	568

### 3.2 Development and Execution Environment

- **Development tools**

e-Climbing software uses such development tools as shown in Table 5.

Table 5. Software development tools		
Windows 10		The system will be developed in the computer where Windows 10 is installed.
NetBeans 8.0 or later		NetBeans 8.0 or later is the IDE for developing e-Climbing. The application use Java as the developing language.
Microsoft Office Visio 2016		The flowcharts, system overview, WBS, DFD will be drawn by using Microsoft Office Visio 2016.
Google Docs		Google Docs was used to create documents and edit with others at the same time.
Google Slides		Google Slides is to create slides and edit with others at the same time.
Google Drive		Google Drive is to share documents and files.
Github		Github is used as the repository for sharing code.
Balsamiq Mockups 3.0		Balsamiq Mockups is used to design UI demo.

- **Development computer**

Table 6. Development computer environment		
Manufacture	Lenovo	DELL
Model	T540P	Inspiron 5558

Processor	4th Gen Intel Core i5-4300M@2.4GHz 4	Inter(R)Core(TM)i3-4030U CPU 1.90GHz
Memory(RAM)	8 GB	6 GB
System Type	64 bit Operating System	64 bit Operating System
Operating System	Microsoft Windows 10	Windows 10 Home
Hard Disk	1TB Serial ATA Hard Drive	452 GB
Monitor	15.6 inch Widescreen Digital Plat Panel	Generic PnP Monitor
Optical Drive	16XDVD+/-RW Drive 12X RAM	P51F-001 DVD
Video Card	GeForce 6150SE nForce 430	Intel(R) HD Graphics Family
Keyboard & Mouse	USB Keyboard and Optical USB Mouse	Pen & touch
Wireless	Internal PCI 802.11g Wireless Network Card	Intel(R) Dual Band Wireless-AC3160
Anti Virus Software	McAfee SecurityCenter with anti-virus, anti-spyware, firewall	McAfee
Cords & Cables	Cat7 Ethernet cable, Power cords	Ethernet Realtek

### 3.3 Programming tools

Table 7. Programming language		
Language	Logo	Feature
Java 8		<ul style="list-style-type: none"> <li>• Popular in industry</li> <li>• Easy to maintain</li> <li>• Ubiquitous</li> </ul>
XML		<ul style="list-style-type: none"> <li>• Easy to use</li> <li>• Increases data availability</li> <li>• Simplifies data sharing</li> </ul>
MySQL		<ul style="list-style-type: none"> <li>• Open source</li> <li>• High availability</li> </ul>

HTML 5		<ul style="list-style-type: none"> <li>• Easily understandable</li> <li>• Easy to maintain and update</li> </ul>
CSS 3		<ul style="list-style-type: none"> <li>• Page formatting</li> <li>• Accessibility</li> </ul>

### 3.4 Naming and Coding Standards

Table 8. Naming and Coding Standards for Java

Type	Rules	Example
Classes	<ul style="list-style-type: none"> <li>• Class names should be nouns, in mixed case with the first letter of each internal word capitalized.</li> </ul>	class PatronRegister
Methods	<ul style="list-style-type: none"> <li>• Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized.</li> <li>• The open and closing bracket should be start at a new line</li> </ul>	<pre>public void generateReport() {</pre>
Variables	<ul style="list-style-type: none"> <li>• Variable names should be short yet meaningful.</li> <li>• Should be avoided of uppercase</li> <li>• One-character variable names should be avoided except for temporary "throwaway" variables.</li> <li>• Common names for temporary variables are i, j, k, m, and n for integer.</li> </ul>	<pre>int i String first_name</pre>
Constants	The names of variables declared class constants should be all uppercase with words.	<pre>static final int STRING_LENGTH = 4;</pre>
Comments	<ul style="list-style-type: none"> <li>• Inline comments: //text</li> </ul>	//first name of the user

	<ul style="list-style-type: none"> <li>Multiple line comments:/*text*/</li> <li>Documentation /**documentation*/</li> </ul>	
Indentation	<ul style="list-style-type: none"> <li>Four spaces should be used as the unit of indentation.</li> <li>Break after a comma.</li> <li>Align the new line with the beginning of the expression at the same level on the previous line.</li> <li>Line wrapping for if statements should generally use the 8-space rule, since conventional (4 space) indentation makes seeing the body difficult.</li> </ul>	<pre>//USE THIS INDENTATION INSTEAD if ((cond1 &amp;&amp; cond2)        (cond3 &amp;&amp; cond4)       !(cond5 &amp;&amp; cond6)) {     doSomething(); }</pre>

Table 9. Naming and Coding Standards for MySQL		
Type	Rules	Example
Lowercase	<ul style="list-style-type: none"> <li>Identifiers should be written entirely in lowercase.</li> </ul>	Use first_name, not "First_Name".
Underscores	<ul style="list-style-type: none"> <li>Object names that are comprised of multiple words should be separated by underscores</li> </ul>	Use word_count or team_member_id, not wordcount or wordCount.
Full words, not abbreviations	<ul style="list-style-type: none"> <li>Object names should be full English words</li> </ul>	Use middle_name, not mid_nm.
Reserved words	<ul style="list-style-type: none"> <li>Avoid using any word that is considered a reserved word in the database that you are using.</li> </ul>	void using words like user, lock, or table.

Table 10. Naming and Coding Standards for XML		
Type	Rules	Example
The underscore (_)	<ul style="list-style-type: none"> <li>The underscores must be used for readability in naming simple elements, with no spaces or hyphens between words;</li> </ul>	<xsd:element name="supplier_name" type="varchar2240"/>

Names	<ul style="list-style-type: none"> <li>Names must not contain special characters.</li> </ul>	Avoid space, ' ', '.', '\$', '%', '#', <, >, &, or ?
Upper Camel case	<ul style="list-style-type: none"> <li>Upper camel case must be used for naming complex types with no spaces, hyphens or underscores between words. The complex type elements must not be more than 22 characters in length.</li> </ul>	<xsd:complexType name="SupplierType"/>
Java keywords	<ul style="list-style-type: none"> <li>Avoid Java keywords as element names.</li> </ul>	abstract, assert, Boolean, break, byte, case, catch, char, class, const, continue, default, do, double, else, enum, extends, false, final, finally, float, for, goto, if, implements, import, instanceof, int, interface, long, native, new, null, package, private, protected, public, return, short, static, strictfp, super, switch, synchronized, this, throw, throws, transient, true, try, void, volatile, while.
Indentation	3 spaces per indent level	<p style="color: red;"><b>&lt;note&gt;</b></p> <p style="color: green;"><b>&lt;to&gt;Tove&lt;/to&gt;</b></p> <p style="color: blue;"><b>&lt;from&gt;Jani&lt;/from&gt;</b></p> <p style="color: blue;"><b>&lt;/note&gt;</b></p>

### 3.5 Exception Handling

e-Climbing will use “try-catch” blocks for exception handling by providing one or more catch blocks directly after the try block. For known errors, specific exception handling methods for them will be defined and implemented. All unknown errors will be handled by a generic exception handler.

Demo for exception handling

```

try {
    statements
} catch (ExceptionType name) {
    System.err.println("ExceptionType " + name.getMessage());
} catch (IOException name) {
    System.err.println("Caught IOException: " + name.getMessage());
}

```

### 3.6 Fault Tolerance

A suitable error message will be displayed to the screen whenever the user gets an error condition.

- Frontend (User Interface)  
More detail will be addressed in test plan document.
- Backend(Database)  
More detail will be addressed in test plan document.

### 3.7 Design Constraints

- **Color**

The e-climbing webpage color follows the consistent use of South Dakota State University's official colors for the purpose of building brand consistency and awareness in the marketplace. The primary visual identifier of webpage is blue as shown in Figure 2.



Figure 2. Primary colors

The secondary color should comply with the university's graphic standards as shown in Figure 3. This range of colors may be used in limited quantities such as highlight text or graphs, but they should never appear as the dominant color in any webpage.

				
PMS 1795 0c 96m 90y 2k 210r 38g 48b #D22630	PMS 185 0c 92m 76y 2k 228r 0g 43b #E4002B	PMS 186 (PRT) 0c 100m 75y 4k 200r 16g 46b #C8102E	PMS 200 (USD) 3c 100m 66y 12k 186r 12g 47b #BADC2F	PMS 416 22c 14m 24y 45k 126r 127g 116b #7E7F74
				
PMS 430 33c 18m 13y 37k 124r 135g 142b #7C878E	PMS 444 38c 15m 18y 43k 113r 124g 125b #717C7D	PMS 542 64c 19m 1y 4k 123r 175g 212b #7BAFD4	PMS 549 59c 8m 9y 19k 107r 164g 184b #6BA4B8	PMS 5493 46c 5m 14y 14k 127r 169g 174b #7FA9AE
				
PMS 123 0c 28m 98y 0k 255r 119g 44b #FFC72C	PMS 012 0c 8m 98y 0k 225r 215g 0b #FFD700	PMS 118 5c 26m 100y 27k 172r 132g 0b #AC8400	PMS 137 0c 38m 95y 0k 255r 163g 0b #FFA300	PMS 110 2c 24m 100y 7k 218r 170g 0b #DAAA00
				
PMS 335 100c 0m 58y 22k 0r 123g 95b #007B5F	PMS 347 (4-H) 89c 0m 90y 0k 51r 153g 102b #339966	PMS 348 100c 4m 87y 18k 0r 132g 61b #00843D	PMS 349 94c 11m 84y 43k 4r 106g 56b #046A38	PMS 359 42c 0m 48y 0k 161r 216g 132b #A1D884
				
PMS 376 53c 0m 96y 0k 132r 189g 0b #84BD00	PMS 377 51c 5m 98y 23k 122r 154g 1b #7A9A01	PMS 383 26c 3m 93y 17k 168r 173g 0b #A8AD00	PMS 389 23c 0m 83y 0k 208r 223g 0b #D0DF00	PMS 575 57c 11m 85y 45k 103r 130g 58b #67823A
				
PMS 7494 31c 5m 36y 16k 156r 175g 136b #9CAF88	PMS 462 28c 48m 71y 72k 92r 70g 43b #5C462B	PMS 464 13c 51m 87y 48k 139r 91g 41b #8B5B29	PMS 471 5c 70m 97y 20k 184r 97g 37b #B86125	PMS 513 56c 98m 0y 0k 147r 50g 142b #93328E
				
PMS 520 66c 86m 5y 17k 100r 47g 108b #642F6C	PMS 525 71c 90m 9y 37k 87r 44g 95b #572C5F	PMS 526 76c 99m 0y 0k 112r 47g 138b #702F8A	PMS 527 75c 100m 0y 0k 128r 49g 167b #8031A7	

Figure 3. Secondary color

- **Global variable**

Any global variable should be documented in the table of GlobleVar.doc file as shown in table 11.

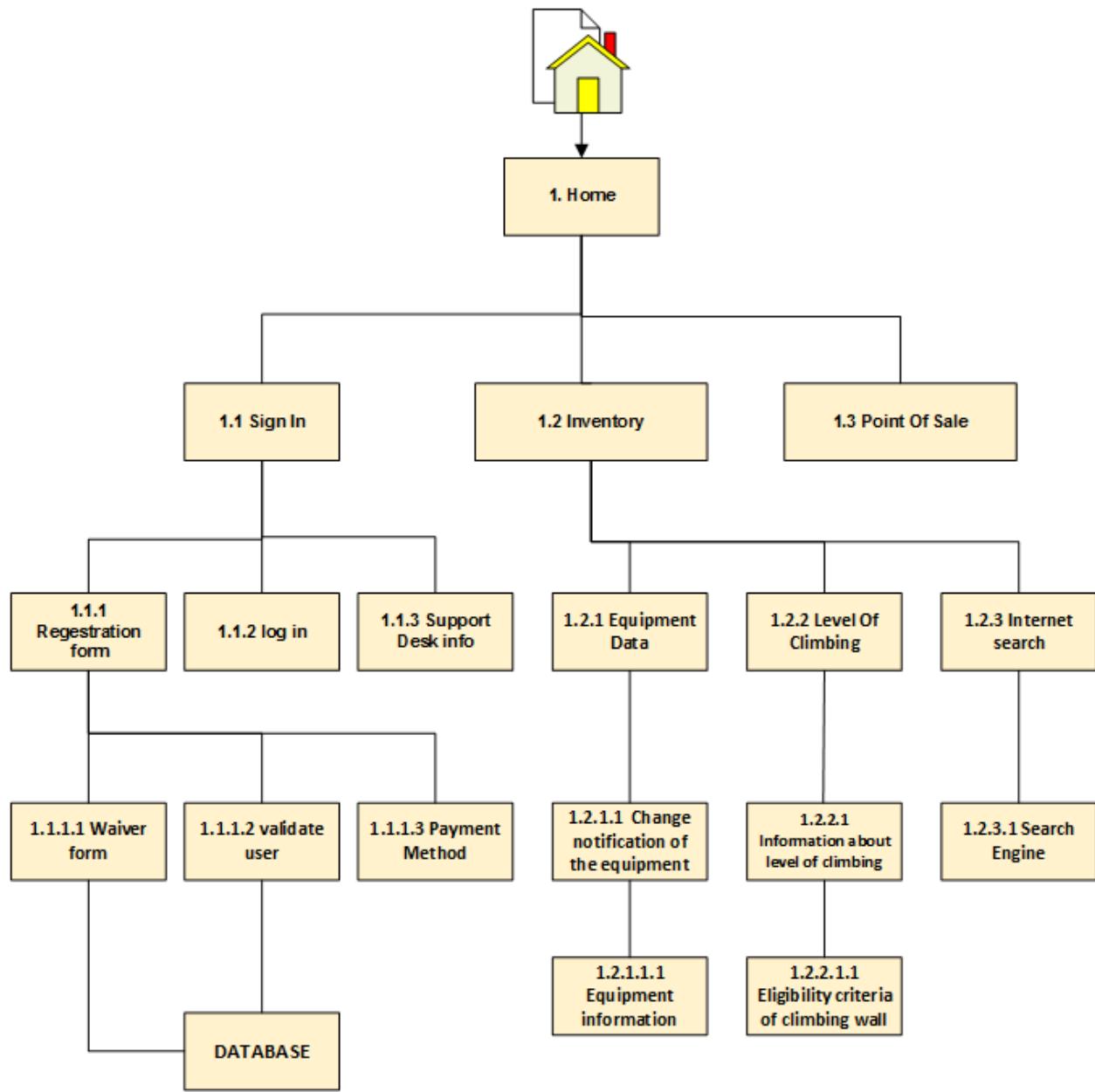
- Any global variable should be uppercase.

- The variable name should be meaningful.

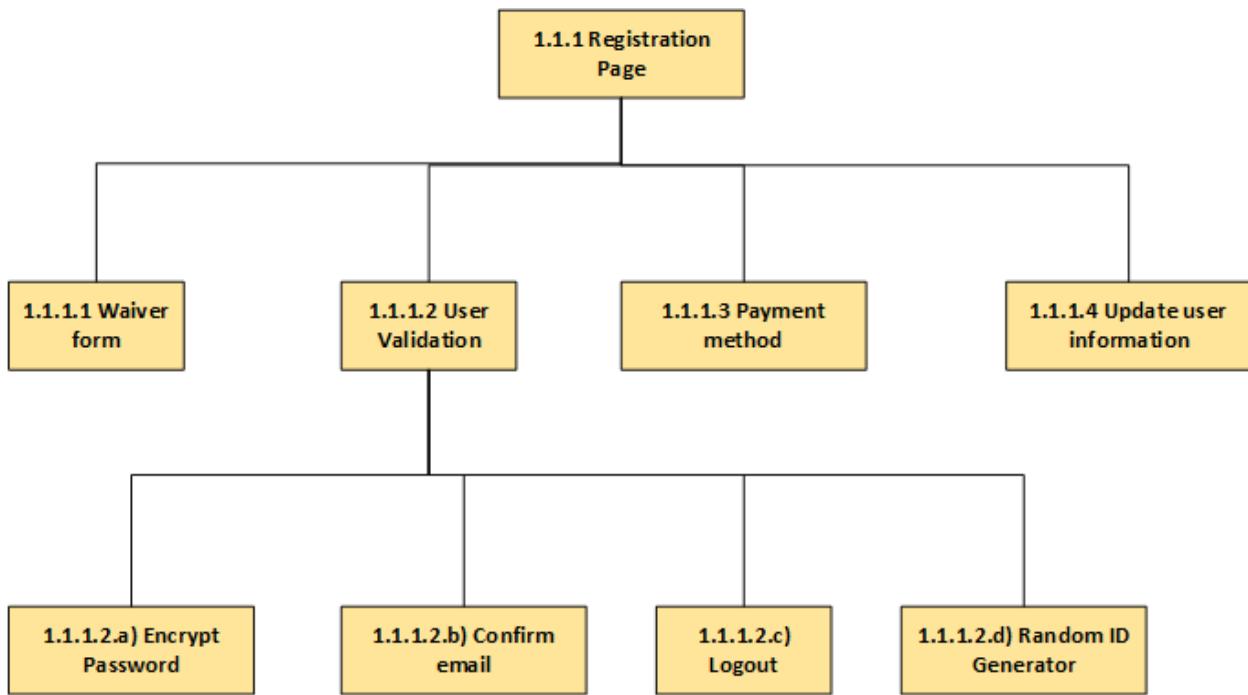
Table 11. Global variable management		
Name	Type	Declared file
NAME_LENGTH	String	Registration.java
XX	XX	XX
XX	XX	XX

## 4. DATA DESIGN

The following Data Flow Diagrams(DFD) and data dictionaries (See Appendix A) demonstrate the data design in e-Climbing system.

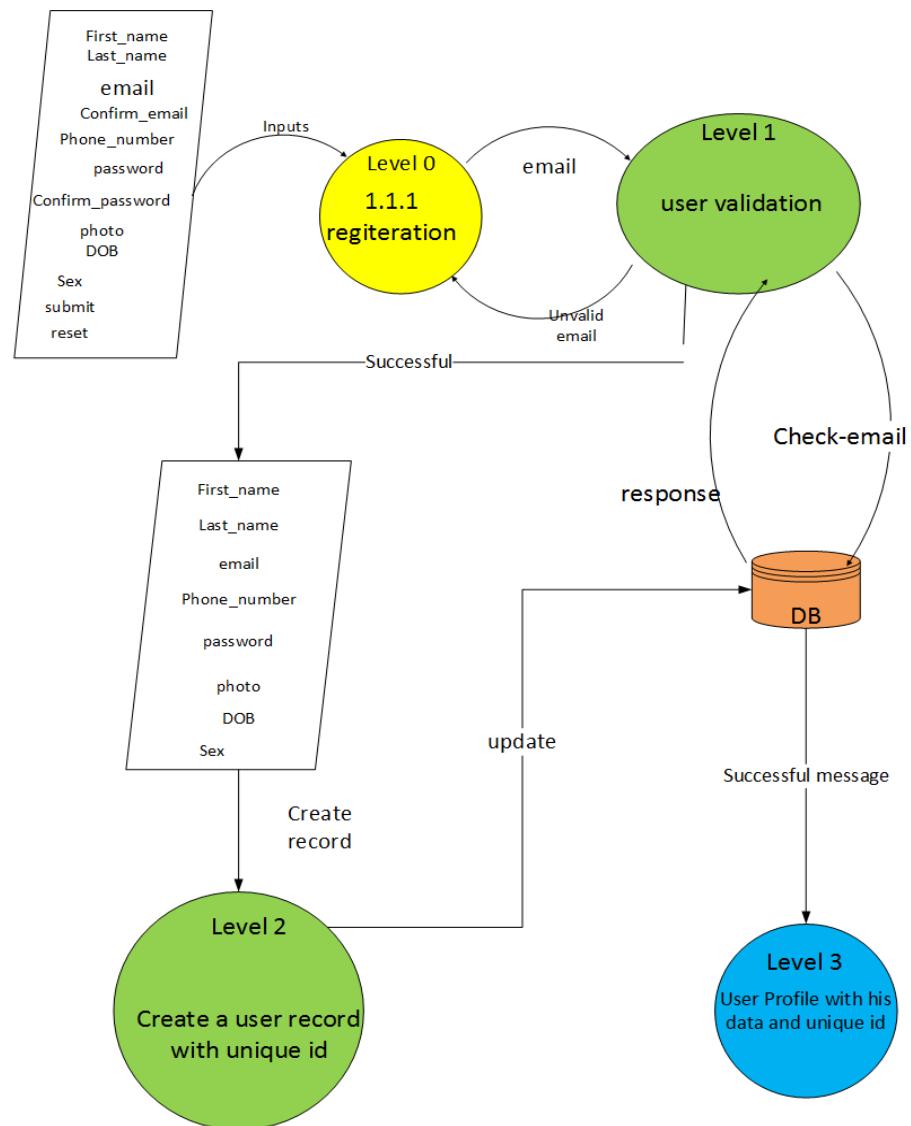


### 1.1.1 Registration of patron



WBS No. 1.1.1 registration	
Description	Allows user to enter the information and validate the information.
Sprint	S1
Developer team	Hussain, Appala, Shaohu
Validation	Valid email address
Input	First_name, last_name, email, confirm_email, password, confirm password, phone, DOB, Sex, and submit action
Output	Successful message with user unique id and his information.

Data flow diagram for registration (WBS NO 1.1.1):



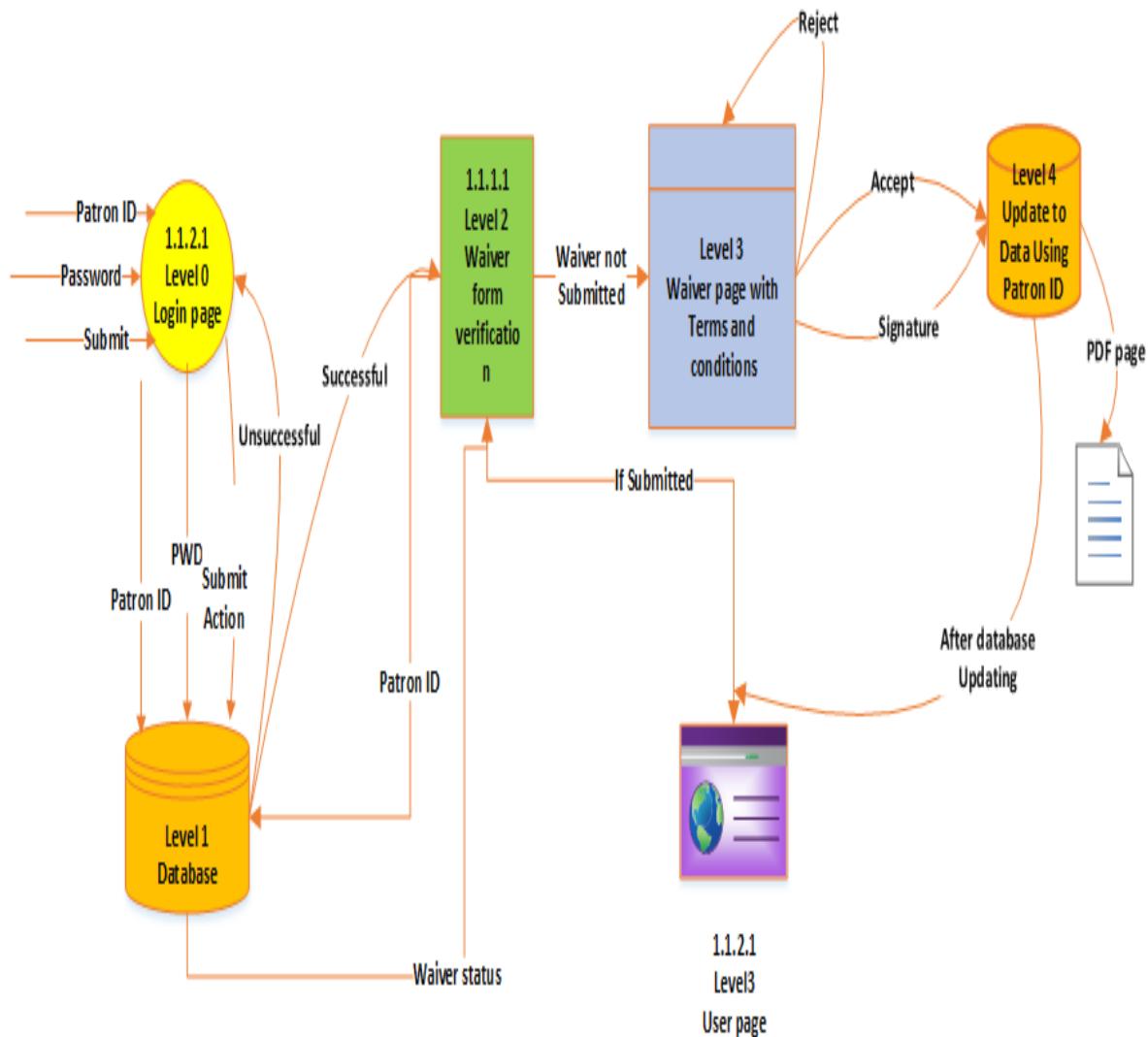
Screenshot for registration (WBS NO 1.1.1):

The screenshot shows a web browser window with the URL <https://www.sdsstate.edu/wellness-center/climbing-wall>. The page has a blue header with the South Dakota State University logo and the text "Wall Climbing Center". Below the header is a form with fields for First Name, Last Name, Password, Confirm Password, Email, Confirm Email, Phone, and User Name. Each field has a yellow placeholder box. At the bottom of the form are "Submit" and "Reset" buttons. The footer contains the university's address (Brookings, SD 57007), phone number (1800.62.3641), and copyright information (Copyright © 2016).

#### 1.1.1.1 waiver form

WBS No. 1.1.1.1 waiver form	
Description	The waiver form will give the description of the risk management policies of the rock climbing center
Sprint	S2
Developer team	Hussain, Appala, Shaohu
Input	Patron name, ID, with the acceptance of the waiver form by the patron
Output	The waiver form of the patron will be updated to the database in the form of the pdf file

Data Flow diagram for Waiver form (WBS NO 1.1.1.1)



Screenshot for Waiver form (WBS NO 1.1.1.1)



### SDSU Climbing Gym Acknowledgement of Risk

**PLEASE NOTE:** This Waiver of Liability, Release, Acknowledgement of Risk, and Indemnification Agreement ("Waiver Agreement") is intended to be, and is, legally binding.

If any aspect of this Waiver Agreement requires clarification, have an SDSU Wellness Center Climbing Gym attendant fully explain it, before signing. By signing the Wellness Center Climbing Gym "Sign-in Sheet(s)", you are agreeing to all terms set forth in this Waiver Agreement. You and/or the person on whose behalf you are signing, are waiving the right to bring any type of action, whether in court or otherwise, to recover compensation or obtain any other remedy for any personal injuries, damages to property, any accident or incident of any type, or death, arising out of or related to your use of the Wellness Center Climbing Gym, its facilities, grounds, climbing walls, exercise areas, equipment, whether the use is supervised or unsupervised. Rock climbing is a sport that has inherent risks. While the SDSU Wellness Center Climbing Gym offers the sport of rock climbing in a controlled environment, there is still an assumed risk of injury to persons using the SDSU Wellness Center Climbing Gym. In agreeing to this Waiver Agreement, I hereby acknowledge, understand, and agree on my behalf, and upon behalf of the person for whom I am signing, that the sport of rock climbing and the use of the SDSU Wellness Center Climbing Gym, its facilities, equipment, climbing walls, classes and/or participating in activities sponsored by the SDSU Wellness Center Climbing Gym has inherent risks. These risks include, but are not limited to any injury of damage resulting from:

Negligence of employees, or the SDSU Wellness Center Climbing Gym. Negligent misuse of the facility, climbing walls, or equipment of the SDSU Wellness Center Climbing Gym; Falling off or impacting against the climbing walls, impact surface, floors, or anything else; Rope abrasion, entanglement or other activities occurring on the premises; Cuts or abrasions resulting from any cause whatsoever; Failure of the climbing walls or equipment, whether inside or outside; Personal health problems, whether mental or physical; Negligence of other climbers, visitors, or observers or persons who may be present in or around the climbing area or facility; and/or Negligence or lack of adequate training or any person(s) who seek to assist with medical or other help either before or after any injury or damage may occur.

By signing the "Sign-in Sheet", I, for myself and for my heirs, next of kin assigns, and personal representatives, hereby agree to and do release, indemnify and hold harmless the State of South Dakota, South Dakota Board of Regents, South Dakota State University, and the SDSU Wellness Center Climbing Gym, and their officers, employees, and agents, and/or volunteer assistants, from any and all injuries and damage which I, or the person upon whose behalf I am signing in, may sustain or incur arising out of or related to my use of the SDSU Wellness Center Climbing Gym, its facilities, grounds, climbing walls, exercise areas, equipment, participation in classes or events, and/or outdoor programs guided by or connected with the SDSU Wellness Center Climbing Gym, whether the use is supervised or unsupervised. I, for myself and for my heirs, next of kin, assigns, personal representatives, and persons upon whose behalf I am signing the "Sign-in Sheet," hereby agree to and release, indemnify and hold harmless the State of South Dakota, South Dakota Board of Regents, South Dakota State University, and the SDSU Wellness Center Climbing Gym, and their agents, employees, offices, and, volunteer assistants, from any and all causes of action, claims for damages or demands whatsoever. THIS WAIVER AGREEMENT IS BINDING EVEN IF THE RELEASED PERSON(S) OR ENTITY(IES) HAVE CAUSED OR CONTRIBUTED TO ANY DAMAGE OR INJURY THROUGH THEIR COLLECTIVE OR INDIVIDUAL NEGLIGENCE.

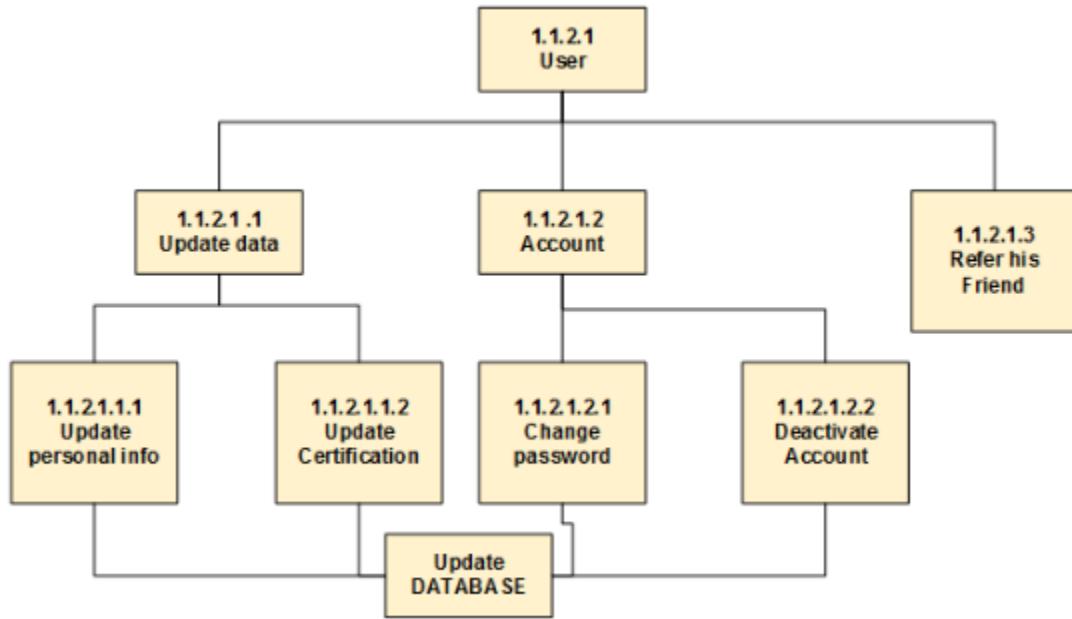
*I and/or person on whose behalf I am signing-in, voluntarily assume complete responsibility for risks and any injuries or damage which may occur as a result of those risks even if the manner or type of injury or damage occurs in a manner that is not foreseeable at the time this Waiver Agreement is accepted. In consideration of my use of the gym, its equipment, employees, volunteer assistants, independent contractors, I agree to and do release, indemnify and hold harmless, the SDSU Wellness Center Climbing Gym, and any and all of their agents, servants and employees, from all liability, claims, demands and damages and further promise not to commence any action or proceeding asserting same.*

All climbers who are twelve (12) years of age or under must be directly supervised by an SDSU Wellness Center Climbing Gym approved adult or be a participant in an SDSU Wellness Center Climbing Gym program. The "Sign-in Sheet" is being signed by the youth's parent, legal guardian, or adult authorized to sign by the youth's parent or legal guardian. By signing the "Sign-in Sheet", the adult acknowledges that they understand the terms of the Waiver Agreement and has the authority to sign for the youth climber. The person signing the "Sign-in Sheet" understands and acknowledges that this Waiver Agreement is binding on the person on whose behalf the "Sign-in Sheet" is signed, for their heirs, next of kin, assigns, and personal representatives.

BY SIGNING the SDSU Wellness Center Climbing Gym "SIGN-IN SHEET" I ACKNOWLEDGE THAT I HAVE READ AND AGREE TO THE TERMS OF THIS WAIVER AGREEMENT. THERE ARE NO ORAL REPRESENTATIONS, STATEMENTS, OR INDUCEMENTS WHICH HAVE BEEN MADE THAT ALTER, CHANGE OR MODIFY ANYTHING SET FORTH IN THIS WAIVER AGREEMENT.

	Print Name	Sign Name	Member Type	Gender: *Optional*	Date & Time Arrived	Time Left
1.						

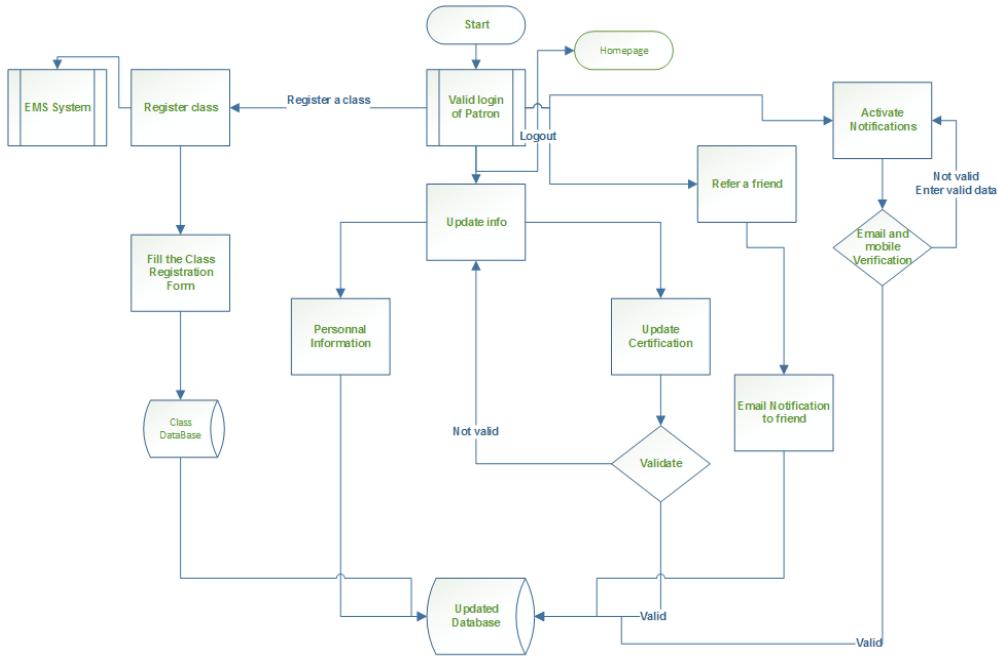
### 1.1.1.2 user validation



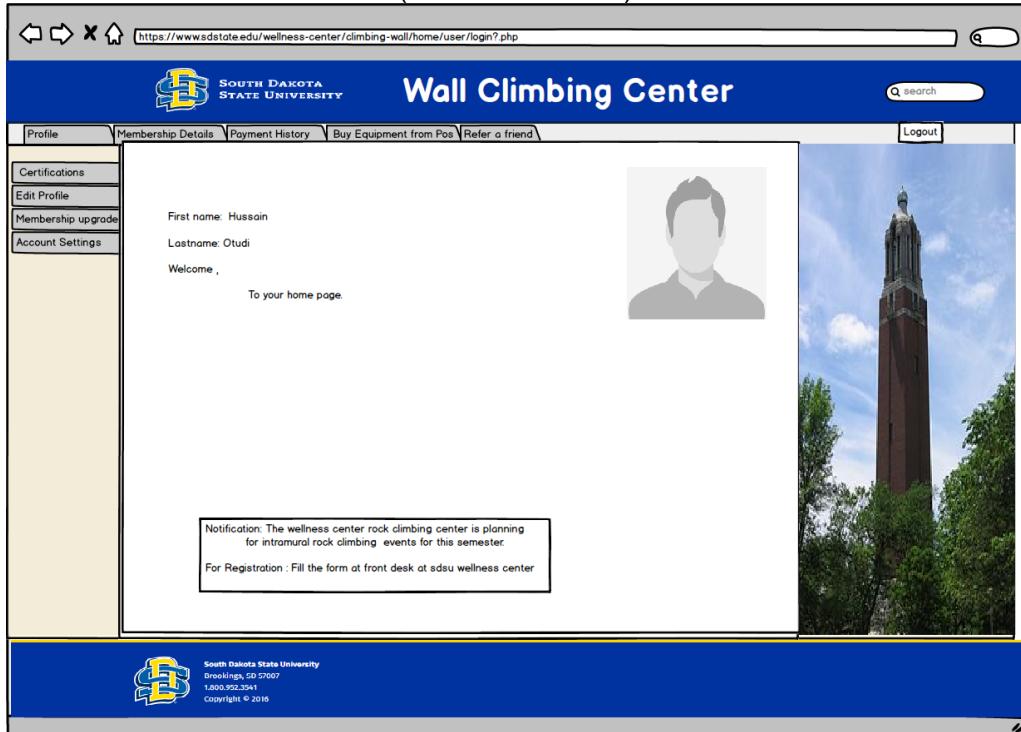
#### WBS No. 1.1.1.2 user validation

Description	Here the Email of the user will get verified in the database in order to remove the duplication creation of the patron account.
Sprint	S3
Developer team	Hussain, Appala, Shaohu
Input	The patron email address will be given as the input for the algorithm.
Output	The email will be verified with the database for the patron before creating the database for the new registered patron.
DFD	See DFD for registration (WBS NO 1.1.1)

Data flow diagram for user validation (WBS NO 1.1.1.2)



screenshot for user validation (WBS NO 1.1.1.2)

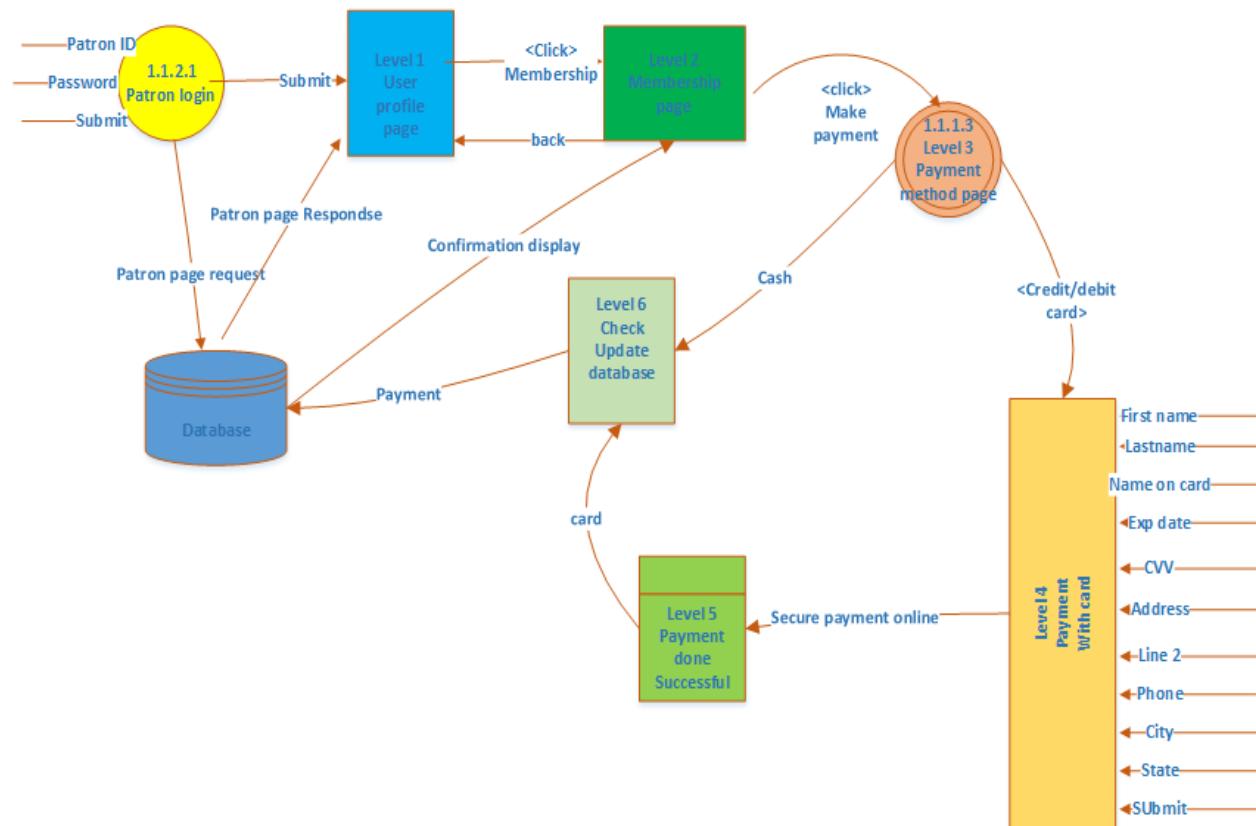


### 1.1.1.3 payment method

#### WBS No. 1.1.1.3 Payment method

Description	Here the payment method can be done for the Membership of the patron from his profile and also for the POS system.
Sprint	S 4
Developer team	Hussain, Appala, Shaohu
Output	The output will be the payment successful/unsuccessful message shown based on the process verification and updated with the Patron ID.

Data flow diagram for the payment process(WBS NO.1.1.1.3):



Screenshot for payment method (WBS NO 1.1.1.3)

[https://www.sdsstate.edu/wellness-center/climbing-wall/Appala/??/Chekuri/Payment.php?/?](https://www.sdsstate.edu/wellness-center/climbing-wall/Appala/??/Chekuri/Payment.php?/)



**SOUTH DAKOTA STATE UNIVERSITY**

## Wall Climbing Center

Pay with a debit or credit card  
If you don't have a PayPal account

Country:

Credit Card number:

Expiration Date:  mm /  yy  CSC  [What's this?](#)

First name:

Last name:

Billing Address Line 1:

Billing Address Line 2 (Optional):

Zip code:

City:

State:

Phone:

Email Address:

[◀ Make Payment](#)

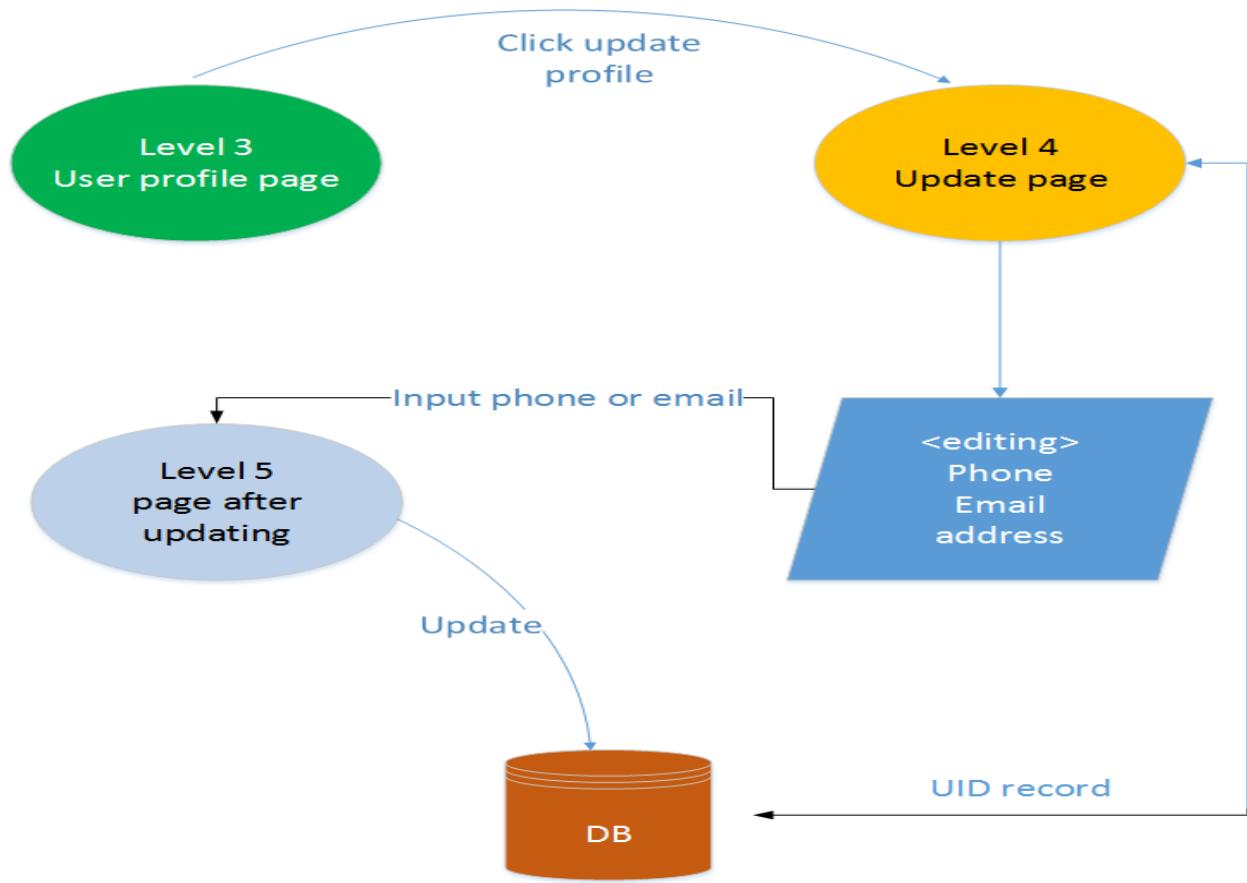


South Dakota State University  
Brookings, SD 57007  
1-800-952-3541  
Copyright © 2016

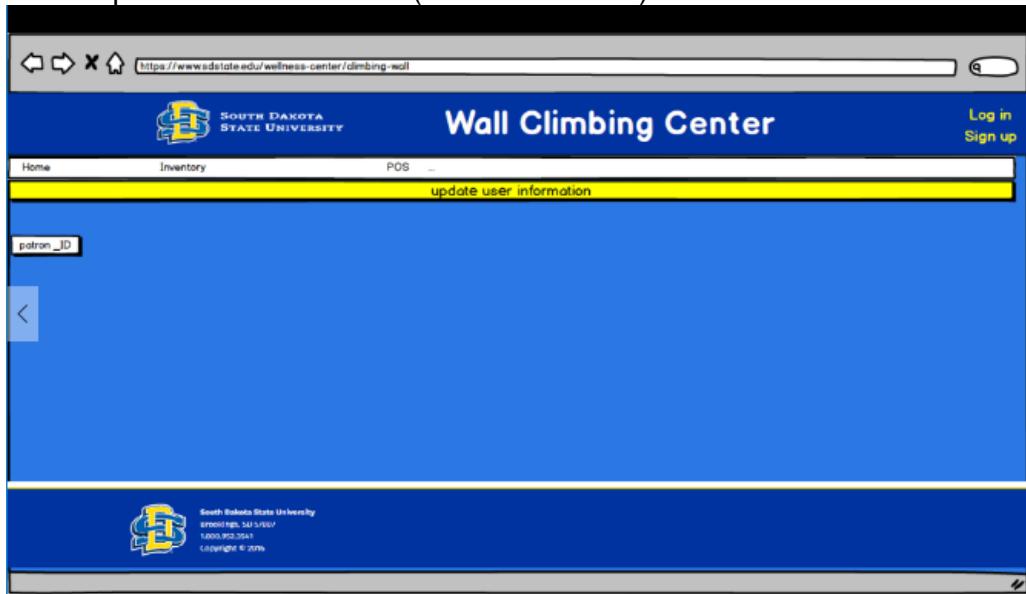
### 1.1.1.4 Update user information

WBS No. 1.1.1.4 update user information	
Description	This function let user update his/her contact information.
Sprint	S5
Developer team	Hussain, Appala, Shaohu
Output	The output will display successful message that his information has been updated

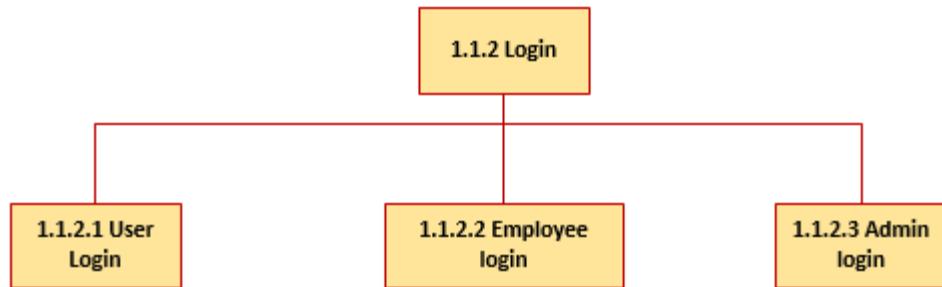
DFD for update user information (WBS NO 1.1.1.4)



Screenshot for update user information (WBS NO 1.1.1.4)

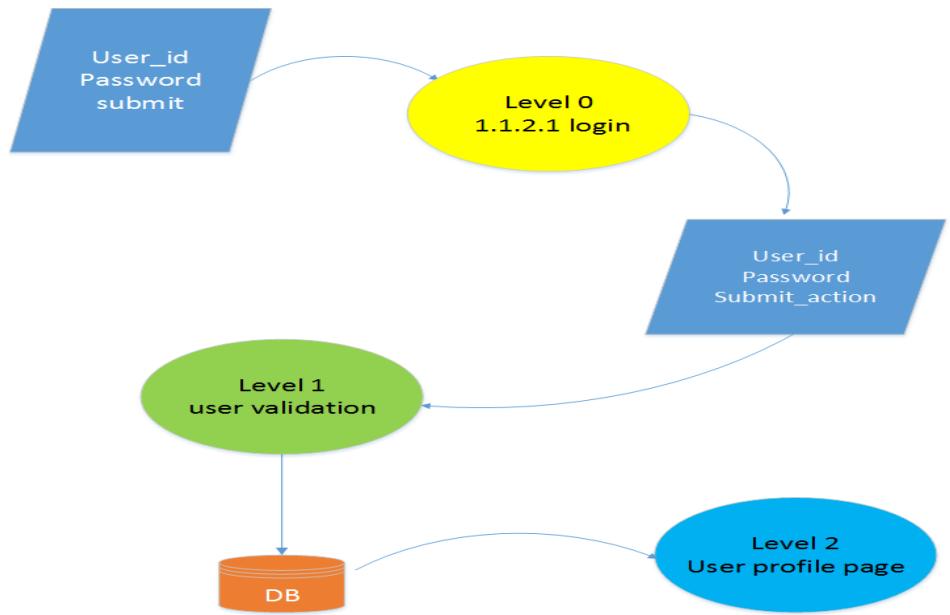


## 1.1.2 Log In

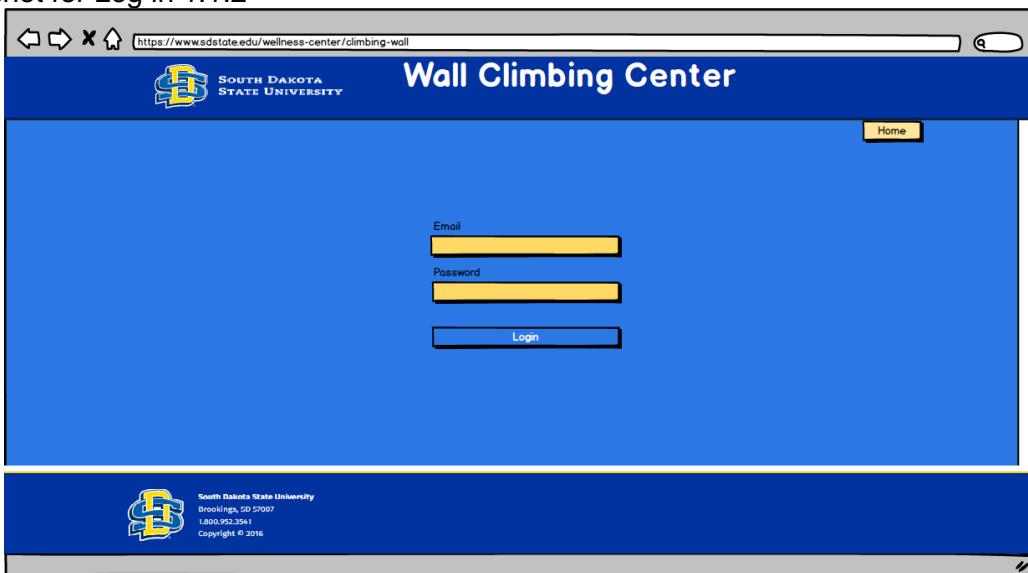


WBS No. 1.1.2 login	
Description	This function for patron login, it will be displayed as html page asking user to enter username and password
Sprint	S6
Developer team	Hussain, Appala, Shaohu
Output	The page will display for user include his/ her name on the top of homepage

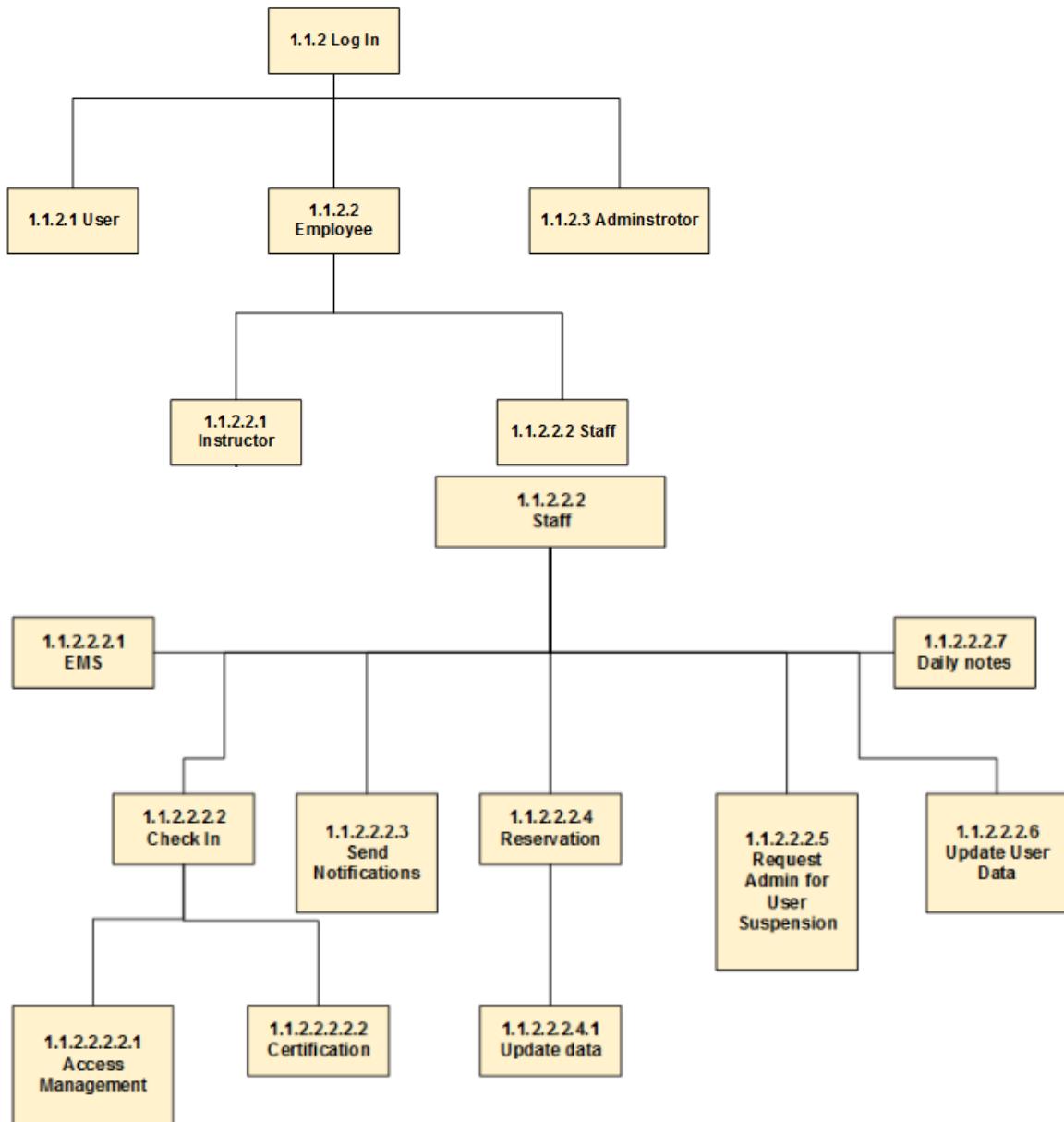
DFD for Patron login (WBS NO 1.1.2)



Screenshot for Log in 1.1.2



### 1.1.2.2 employee

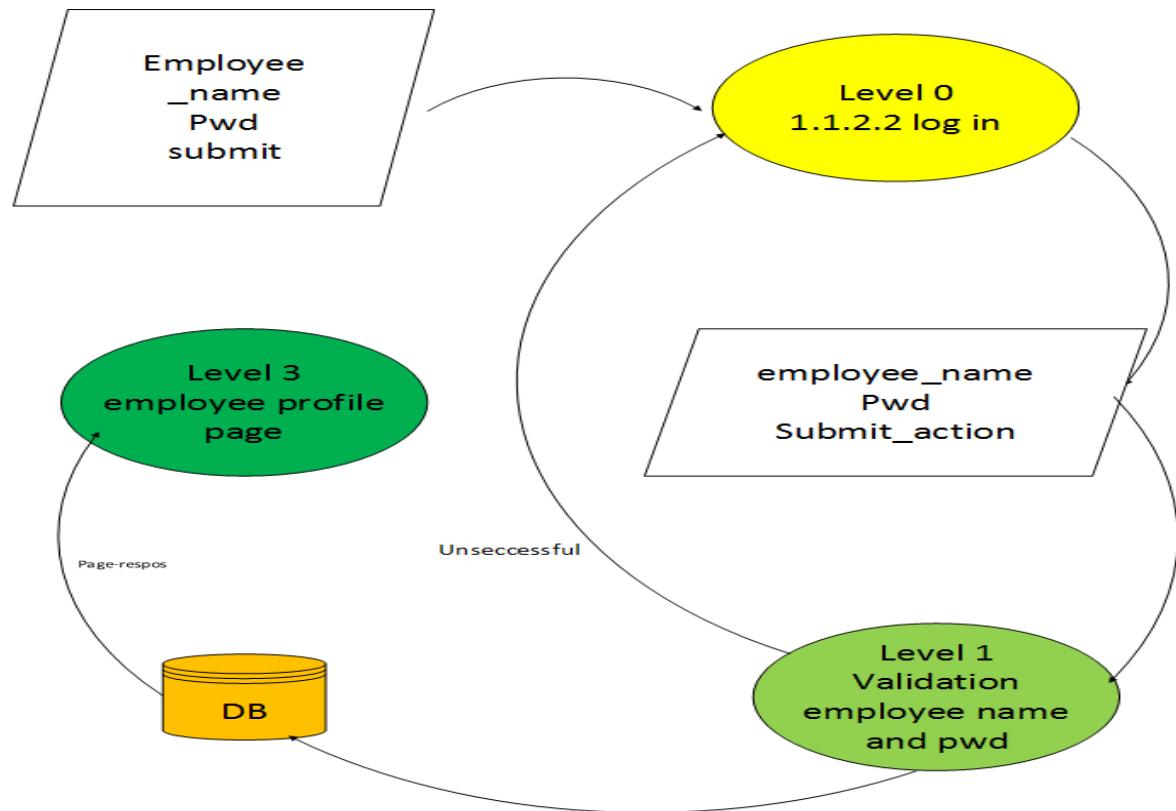


### WBS No. 1.1.2.2 employee

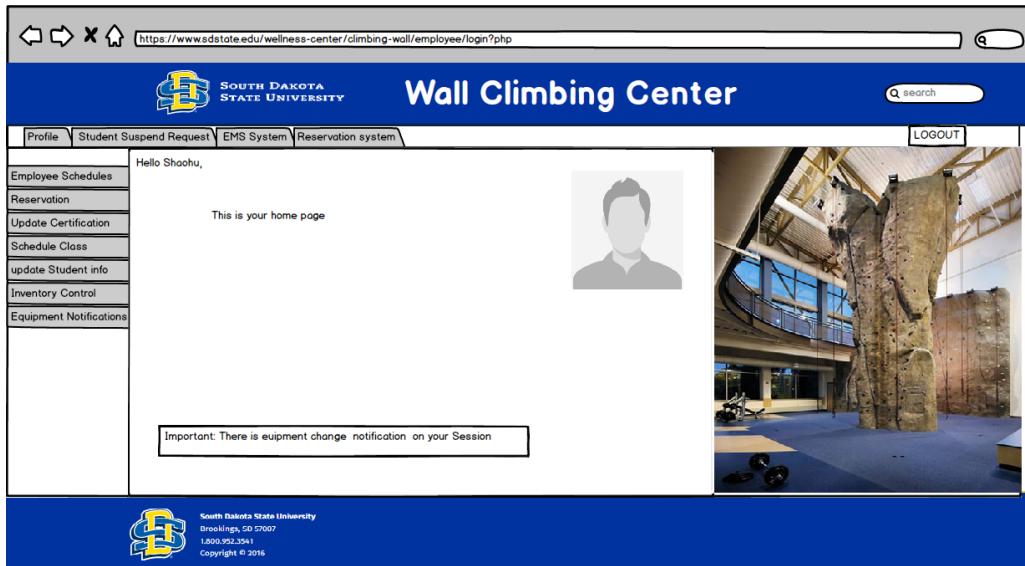
Description	This function for employee login, will be displayed as in html page asking employee to enter his/her name and pwd
Sprint	S7
Developer team	Hussain, Appala, Shaohu

Output	The employee page will be displayed with its details
--------	--

DFD for employee(WBS NO 1.1.2.2)



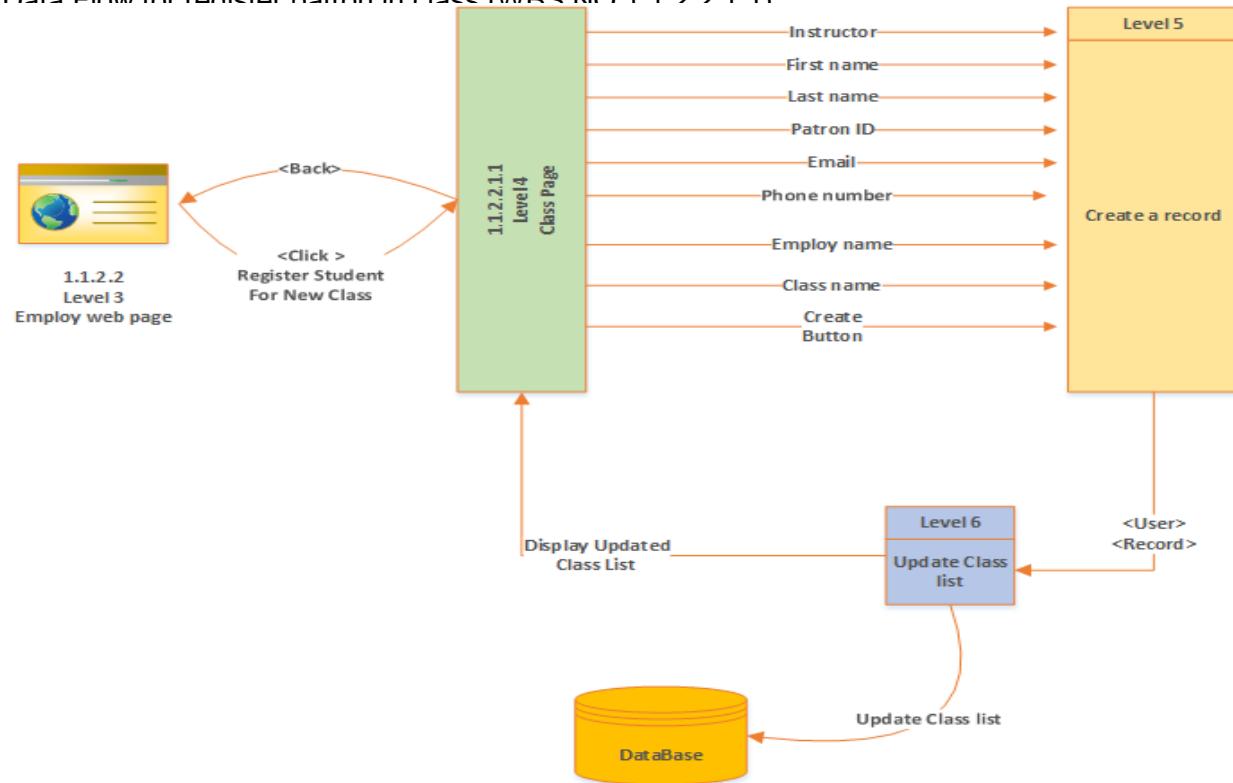
Screenshot of employee (WBS NO 1.1.2.2)



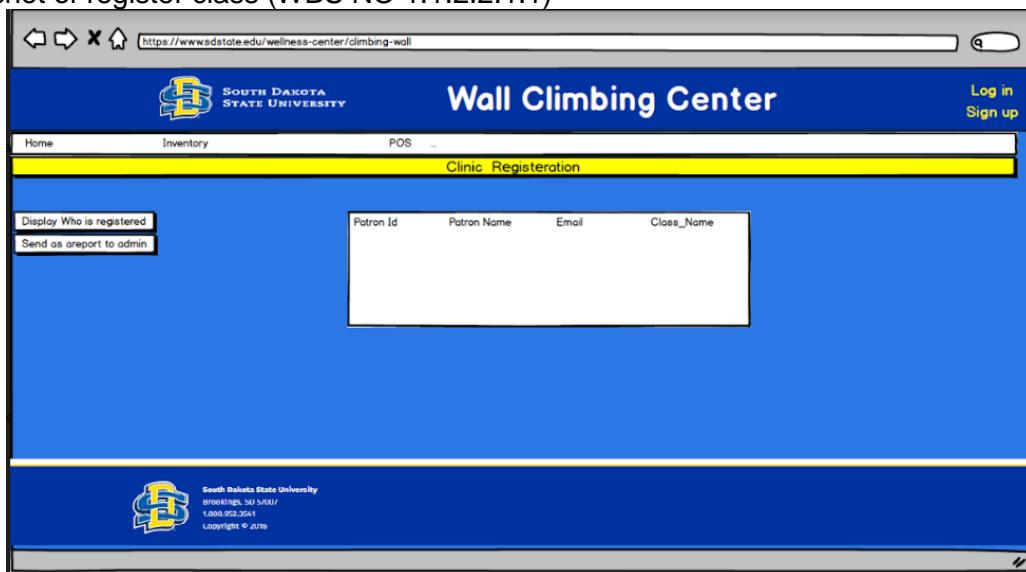
### 1.1.2.2.1.1 Register class

WBS No. 1.1.2.2.1.1 Registering student to a class for certification	
Description	Here the employ/ staff will register the student into the class for certification
Sprint	S8
Developer team	Hussain, Appala, Shaohu
Output	The patron with all his/her details will be created a record and then updated it to class list.

#### Data Flow for register patron in class (WBS NO 1.1.2.2.1.1)



#### Screenshot of register class (WBS NO 1.1.2.2.1.1)

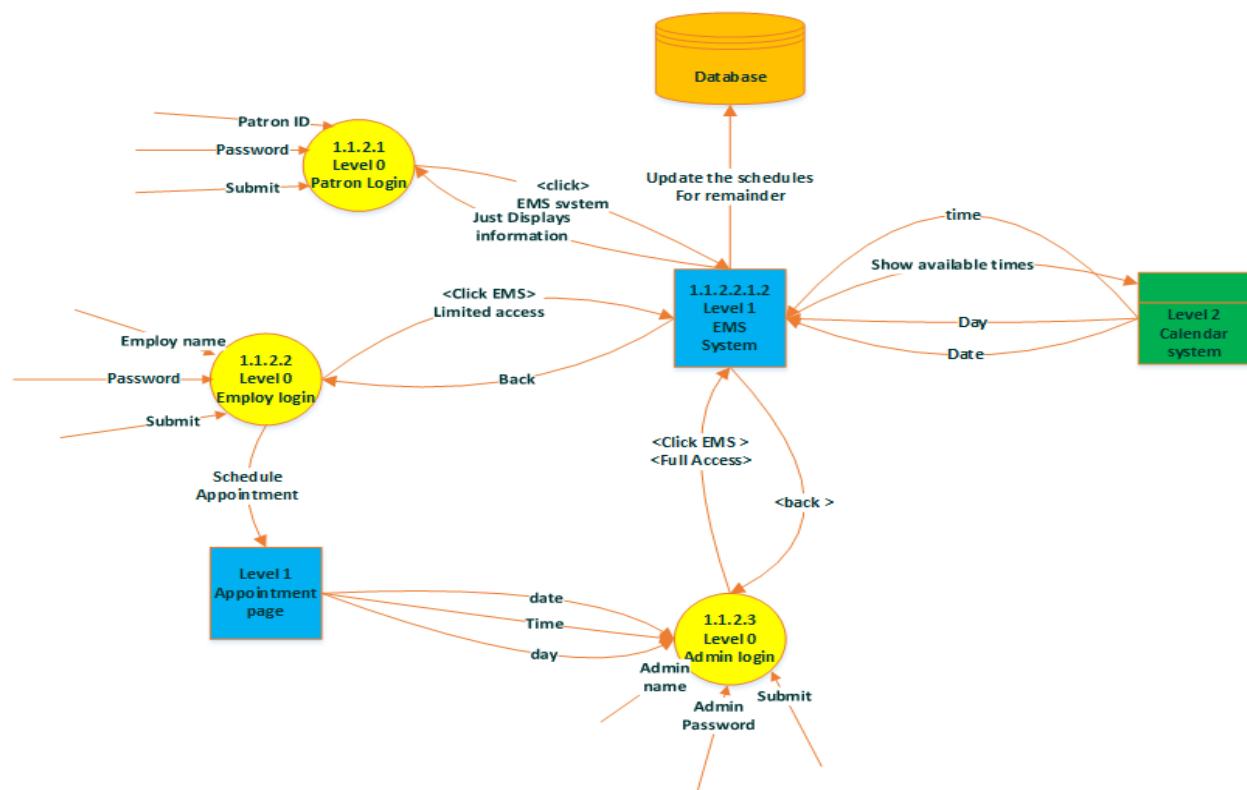


#### 1.1.2.2.1.2 EMS system

WBS No. 1.1.2.2.1.2 EMS System

Description	Display Scheduling System for appointment, Class Timings and the notification remainder and soon. The admin will have the full access to the EMS system.
Sprint	S9
Developer team	Hussain, Appala, Shaohu
Output	The available schedules will be the limited access available timings for the employee for the class timings, notification reminders and that will be displayed on the patron page.

Data flow diagram for the EMS system (WBS NO 1.1.2.2.1.2)



Screenshot for EMS system (WBS NO 1.1.2.2.1.2)

## Details

Add a title for the event

Add a location

Start  
Mon 10/24/2016  9:00 AM   All day

End  
Mon 10/24/2016  9:30 AM   Private

Repeat  Save to calendar  
Never  Calendar

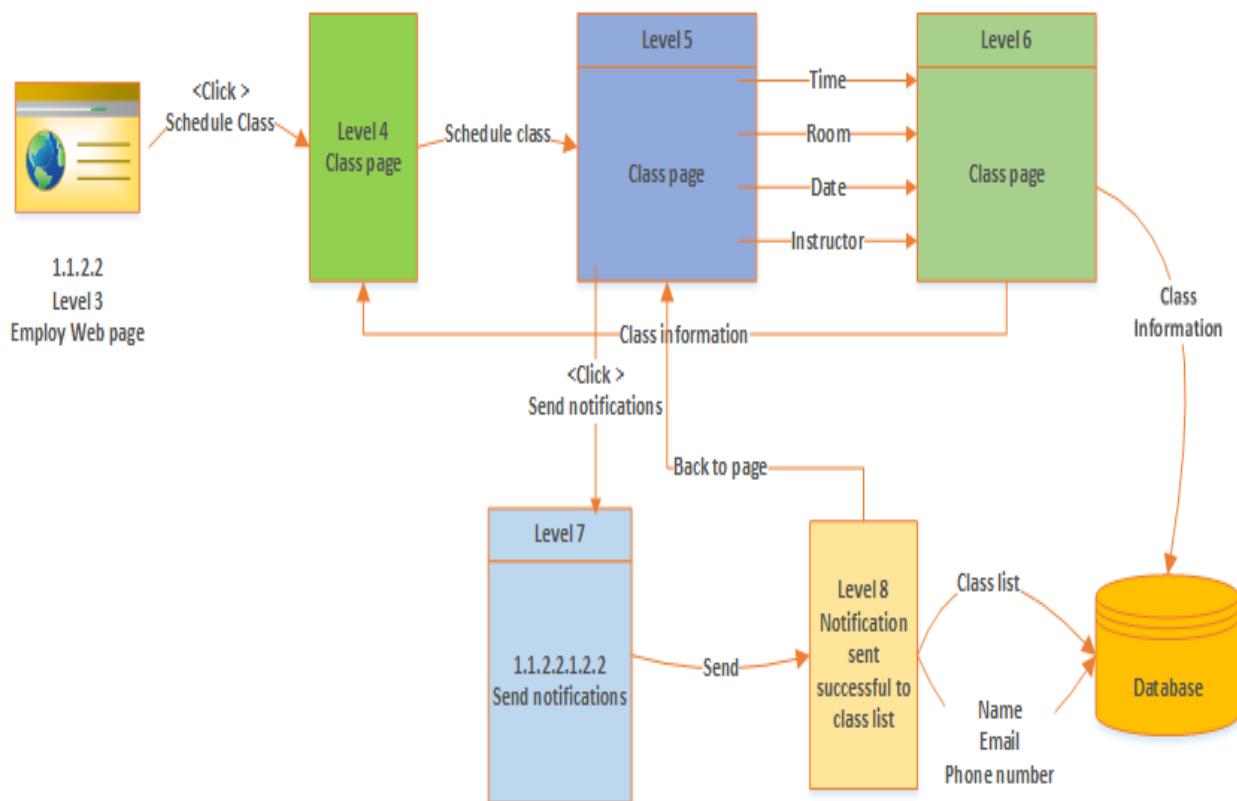
Reminder  Show as  
15 minutes  Busy

[Add an email reminder](#)

### 1.1.2.2.1.2.1 Schedule class for certification

WBS No. 1.1.2.2.1.2.1 Schedule class for certification	
Description	The Scheduling function will show the timing availability for the staff in order to schedule the class for the certification and display it on the user profile .
Sprint	S10
Developer team	Hussain, Appala, Shaohu
Output	The scheduled class information will be updated on the database and will be displayed on the patron screen as notification.

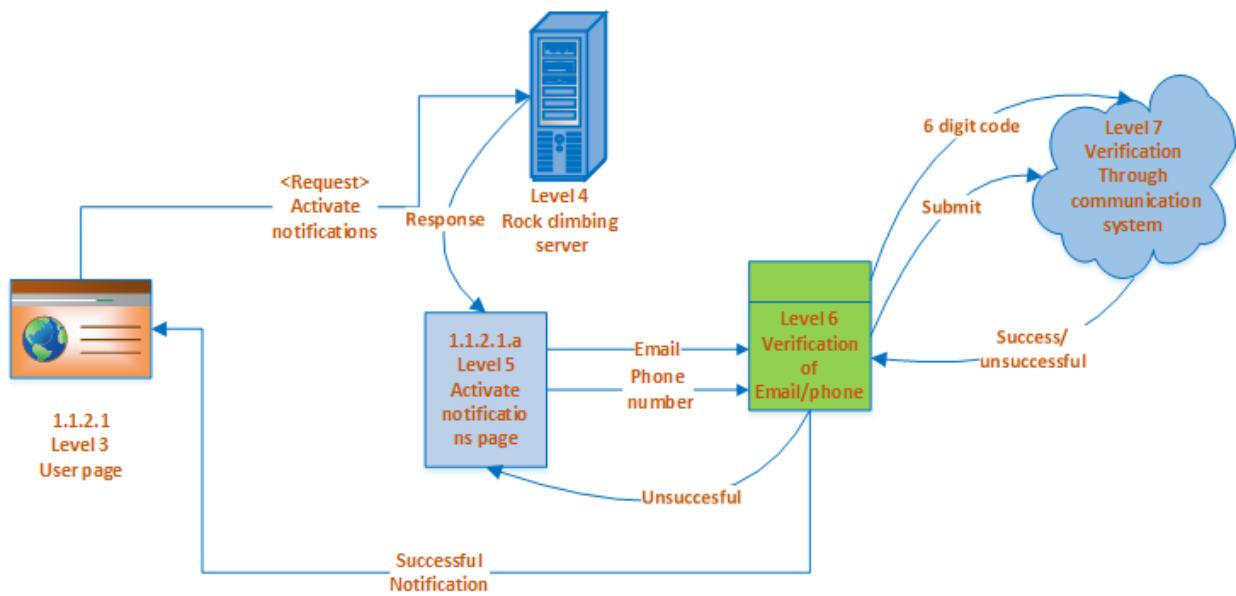
Data Flow diagram for Scheduling class for the EMS system (WBS NO 1.1.2.2.1.2.1)



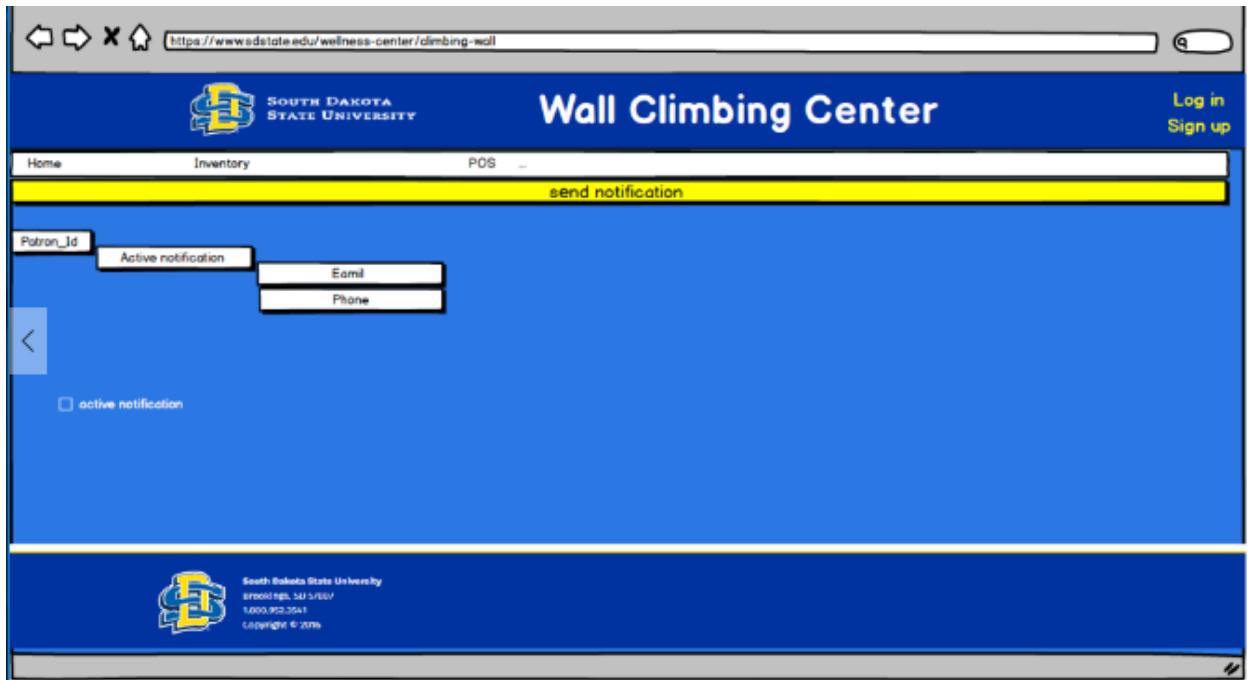
### 1.1.2.2.1.2.2 Send Notifications

WBS No. 1.1.2.2.1.2.2 Send notifications	
Description	The send notifications will be done by the employee to the user page profile. Before that the user has to activate the notifications by valid verification of his email and the phone number of the user.
Sprint	S11
Developer team	Hussain, Appala, Shaohu
Output	The output will be the activate notification of the user successfully on the profile page.

Data flow diagram for the sending notifications (**WBS NO 1.1.2.2.1.2.2**)



Screenshot to active sending notification function (WBS NO 1.1.2.2.1.2.2)

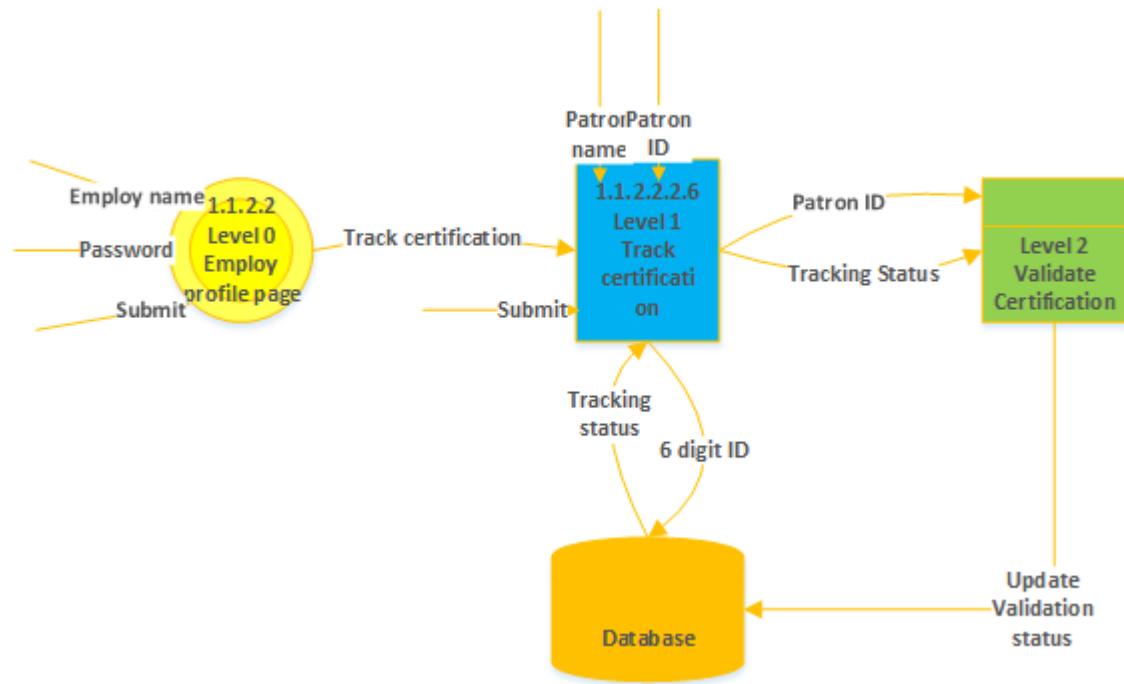


### 1.1.2.2.1.3 certification management

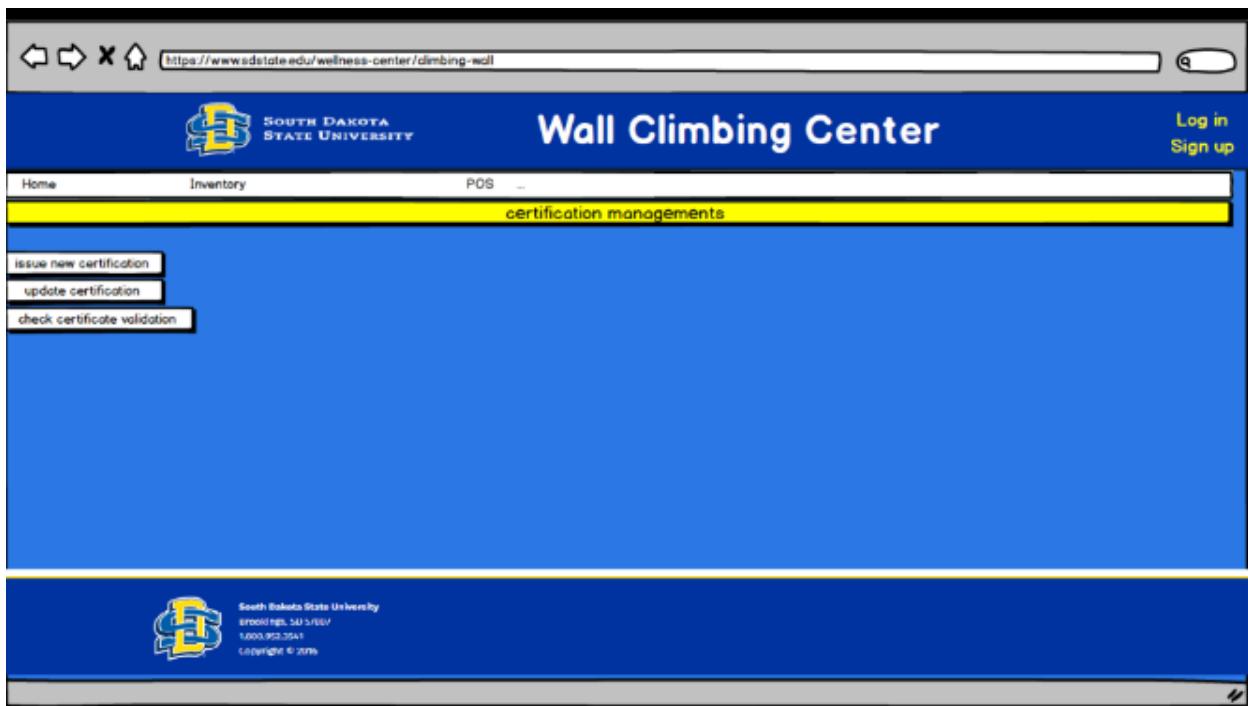
WBS No. 1.1.2.2.1.3 certification management

Description	The certificate management will have the Track certification, validate certification and update validation status of the certification to the database by the employ
Sprint	S12
Developer team	Hussain, Appala, Shaohu
Output	The certification status of the patron will be updated after the validation.

Data Flow for Certificate management:



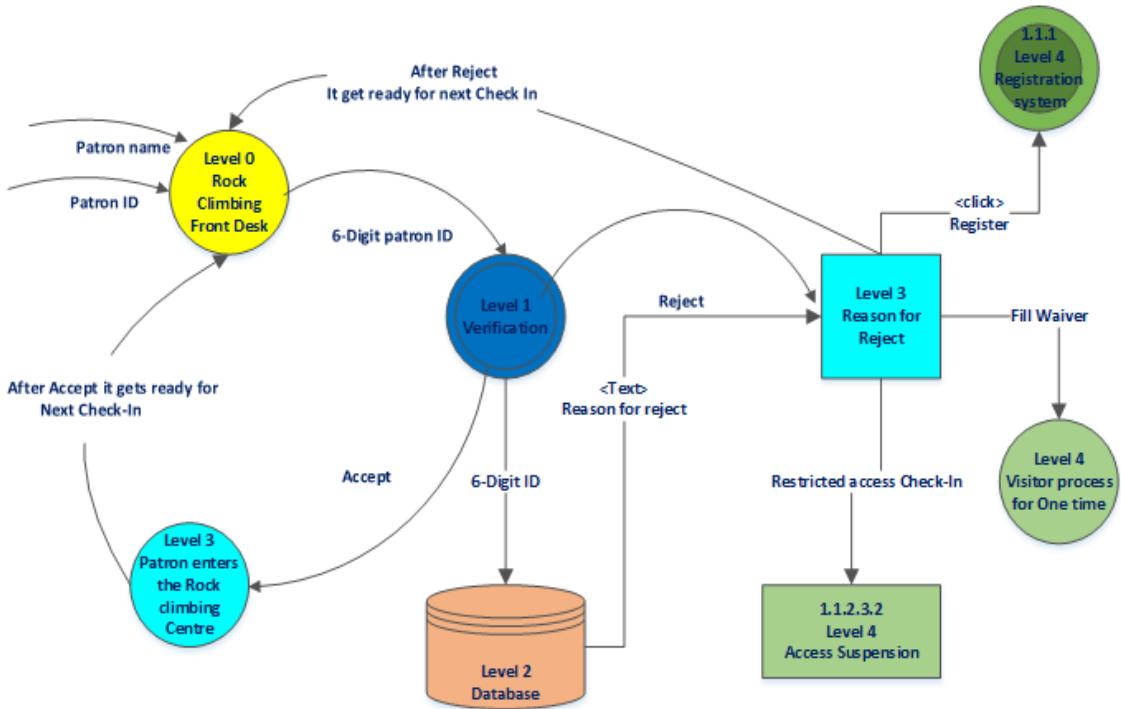
Screenshot for certification management (WBS NO 1.1.2.2.1.3)



#### 1.1.2.2.2.2 check in management

WBS No. 1.1.2.2.2.2 Check In	
Description	The check- in system has very high priority for this system. Here the check in system will check whether the user/ patron has the access to the wall climbing center or not based on his Patron ID.
Sprint	S13
Developer team	Hussain, Appala, Shaohu
Input	patron ID
Output	The output will be based on his Access to the rock climbing system. If it was accepted then he can enter the rock climbing center. If it was rejected then he will get the reason for reject.

Data Flow Diagram for Check In System (WBS NO 1.1.2.2.2.2)



Screenshot for check in management (WBS NO 1.1.2.2.2)

FIRST NAME :	APPALA
LAST NAME :	CHEKURI
DATE OF BIRTH :	MM/DD/YYYY
SEX :	MALE
EMAIL ADDRESS :	XYZ@GMAIL.COM
PHONE :	321-312-XXXX
PAYMENT :	DONE
START DATE :	08/14/2016
END DATE :	12/15/2016
ACCESS :	ENABLE

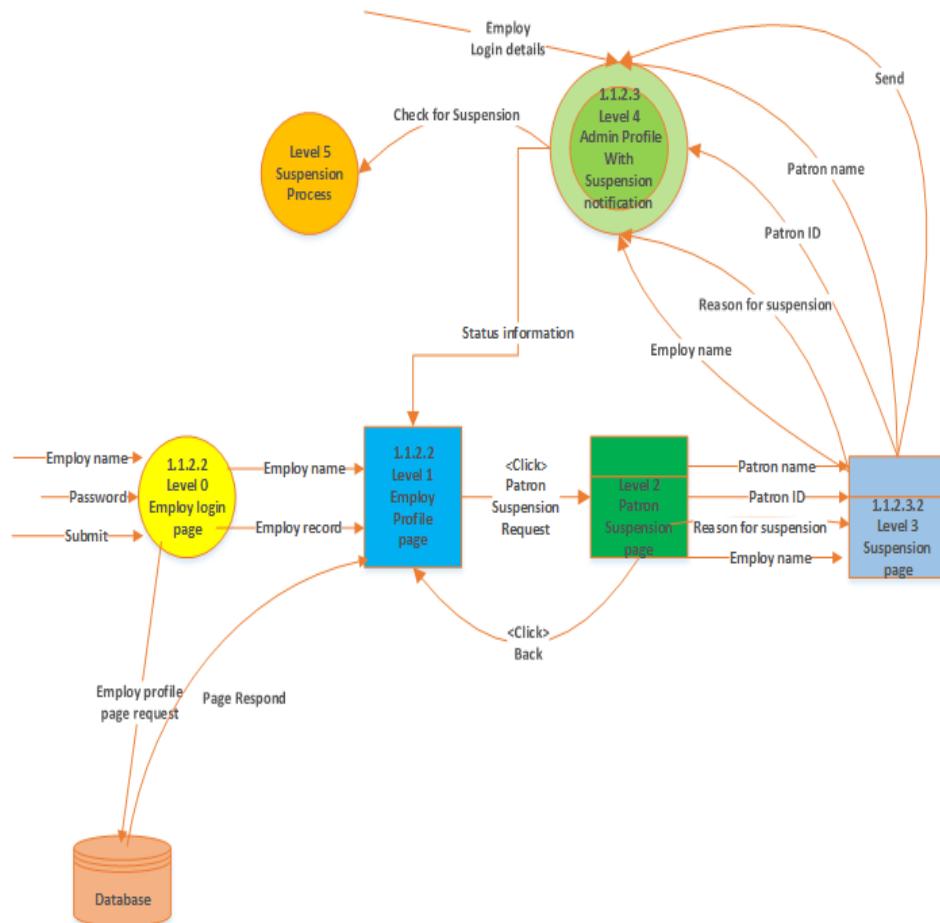
South Dakota State University  
Brookings, SD 57007  
1.800.952.3541  
Copyright © 2016

#### 1.1.2.2.5 request admin for user suspension

### WBS No. 1.1.2.2.5 request admin for user suspension

Description	This function will do process the employee to request user for the patron suspension based on the not following the policies Suspension
Sprint	S14
Developer team	Hussain, Appala, Shaohu
Input	Will be user id and user name and employee who request the suspension and reason of suspension
Output	The output will be approval and do process of suspension or reject the suspension request to employee

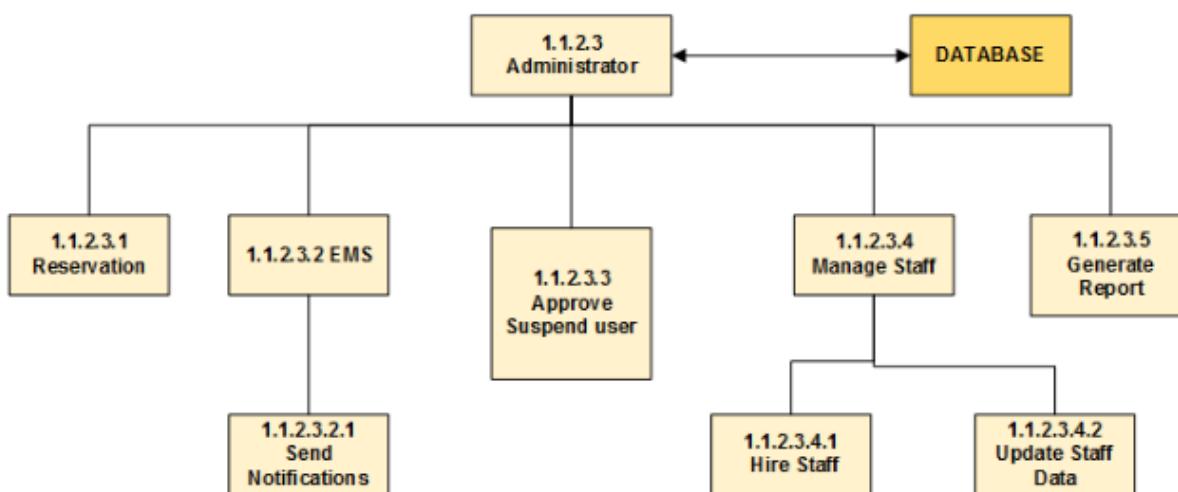
#### Data flow of suspension process (WBS NO 1.1.2.2.5)



Screenshot of suspension process (WBS NO 1.1.2.2.5)

The screenshot shows a web browser window for the "Wall Climbing Center" at <https://www.sdsstate.edu/wellness-center/climbing-wall>. The page features the South Dakota State University logo and navigation links for Home, Inventory, POS, and Request suspension. A central form is displayed with fields for Patron ID, Patron Name, and Reason of Suspension. Below the form is a button labeled "Send a report to Admin". The footer contains the university's logo and contact information.

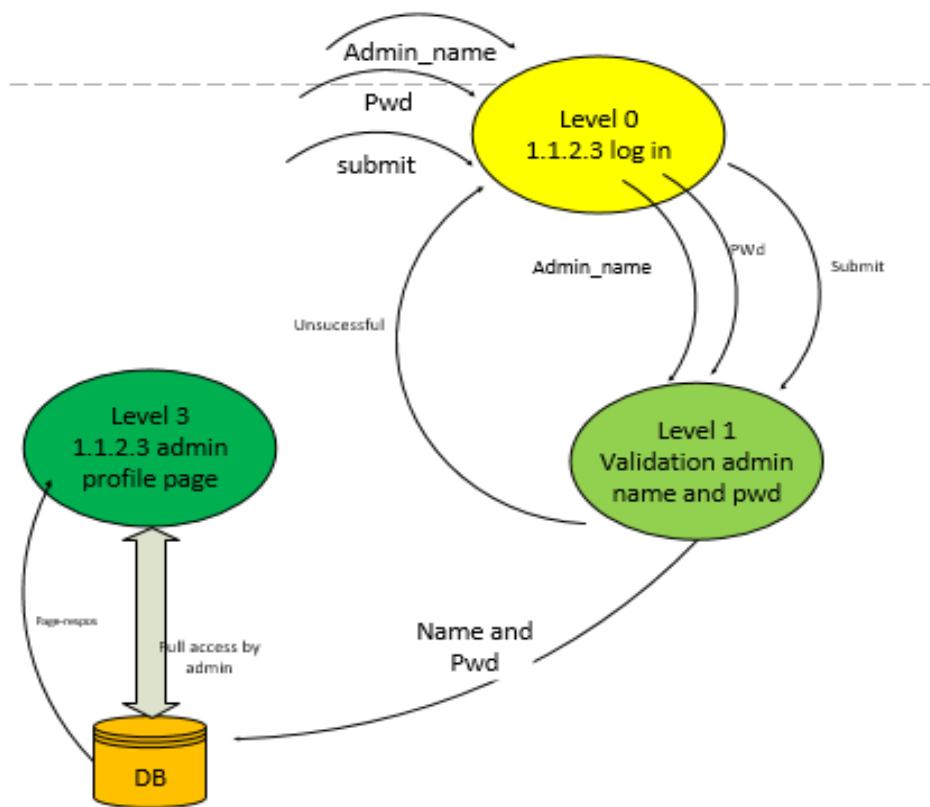
### 1.1.2.3 admin



WBS No. 1.1.2.3 admin

Description	The admin Page has full access for the system. He can remove or add anyone to the system and generate weekly report or whatever he want to modify he can change it
Sprint	S15
Developer team	Hussain, Appala, Shaohu
Output	The admin profile page will be opened and the admin will get the full access to the system of rock climbing.

Data Flow diagram for the admin(WBS NO 1.1.2.3):

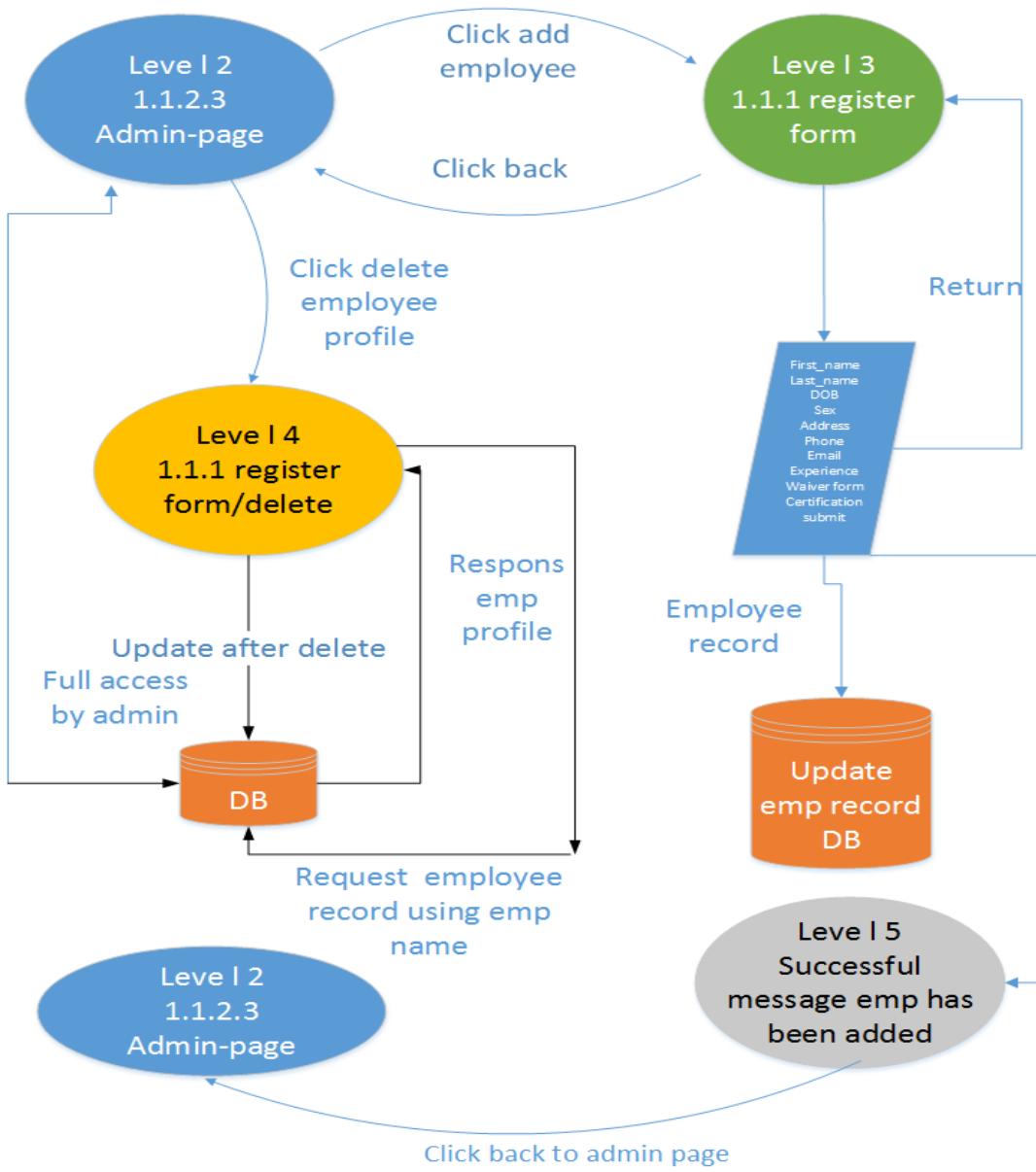


Screenshot for the admin (WBS NO 1.1.2.3):

### 1.1.2.3.1 staff management

WBS No. 1.1.2.3.1 Staff management	
Description	This function will let admin to add or delete an employee
Sprint	S16
Developer team	Hussain, Appala, Shaohu
Output	A message of that indicate the addition or deletion of an employee

Data flow diagram for staff management (WBS NO 1.1.2.3.1)

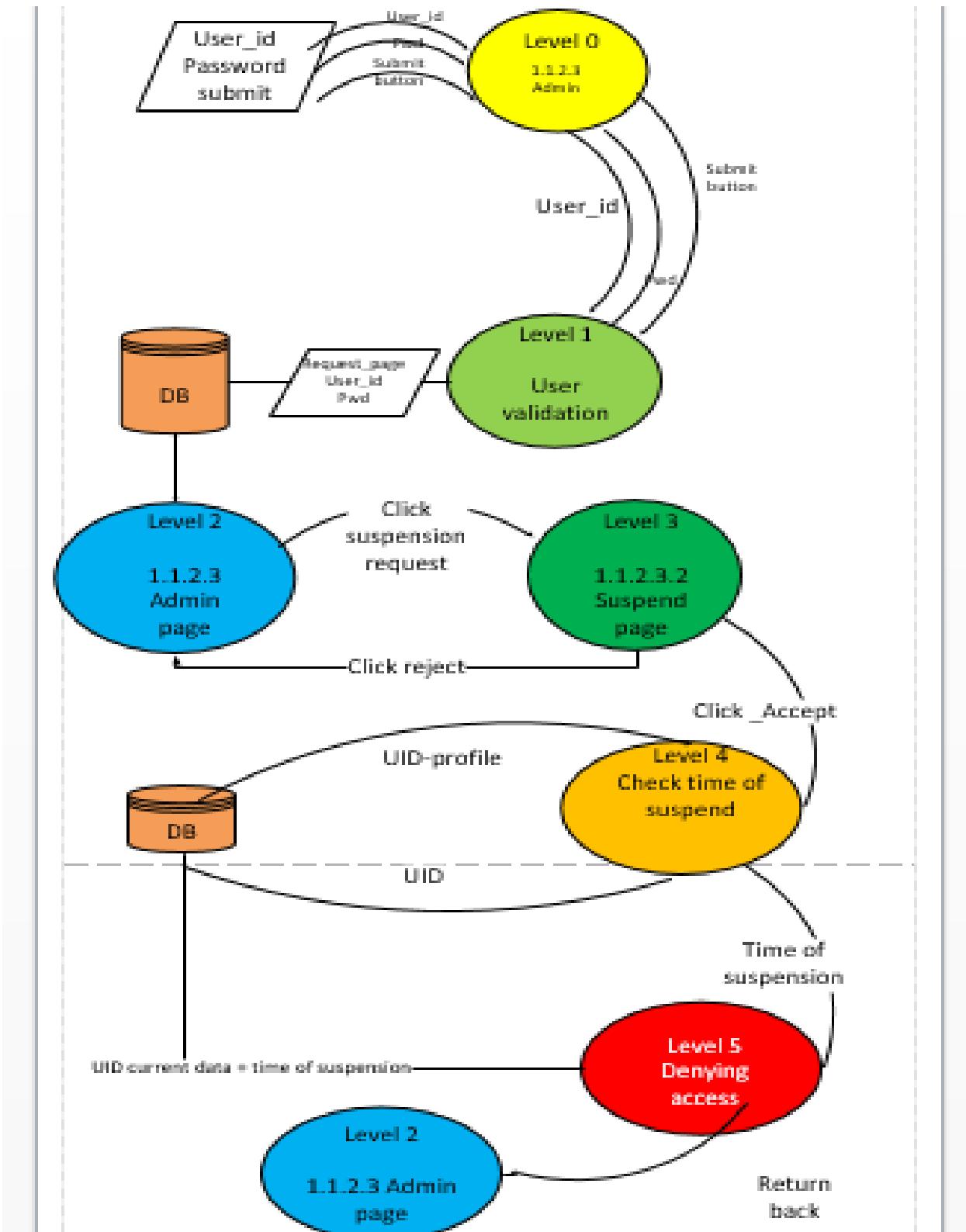


Screenshot for staff management (WBS NO 1.1.2.3.1)

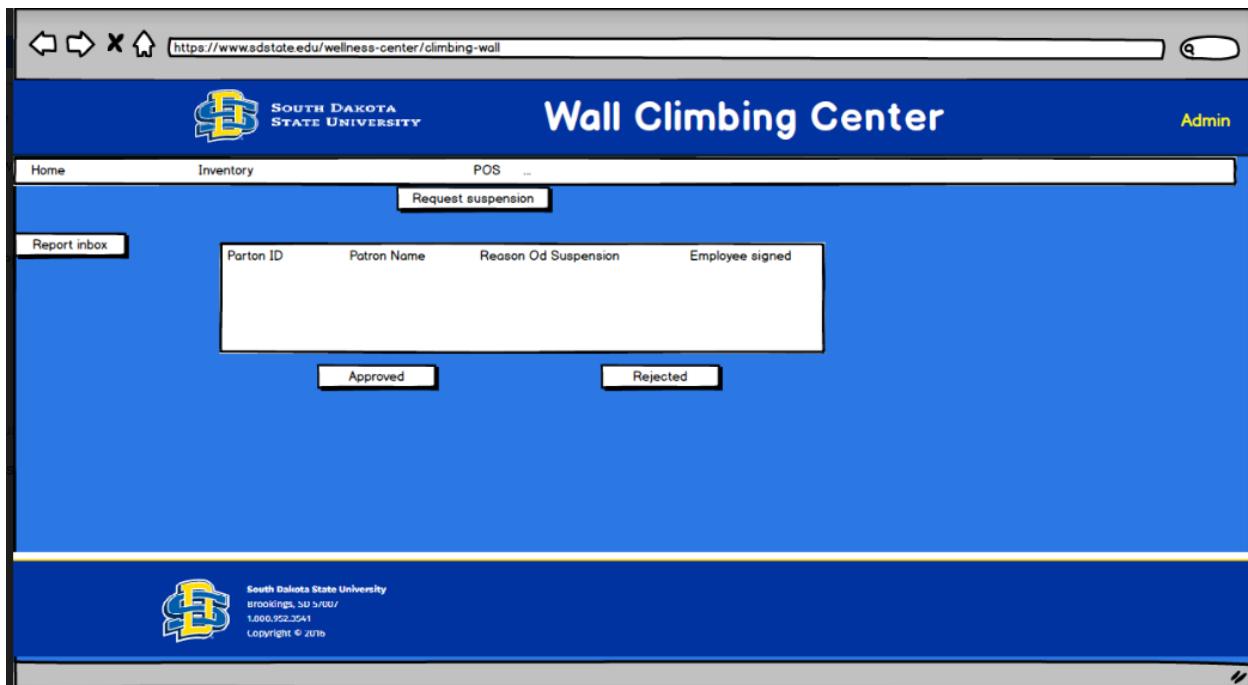
### 1.1.2.3.2 Admin for user suspension

WBS No. 1.1.2.3.2 admin for user suspension	
Description	This function will do process of user suspension
Sprint	S17
Developer team	Hussain, Appala, Shaohu
Output	The output will be approval and do process of suspension or reject the suspension request to employee

Data flow of suspension process (WBS NO 1.1.2.3.2)



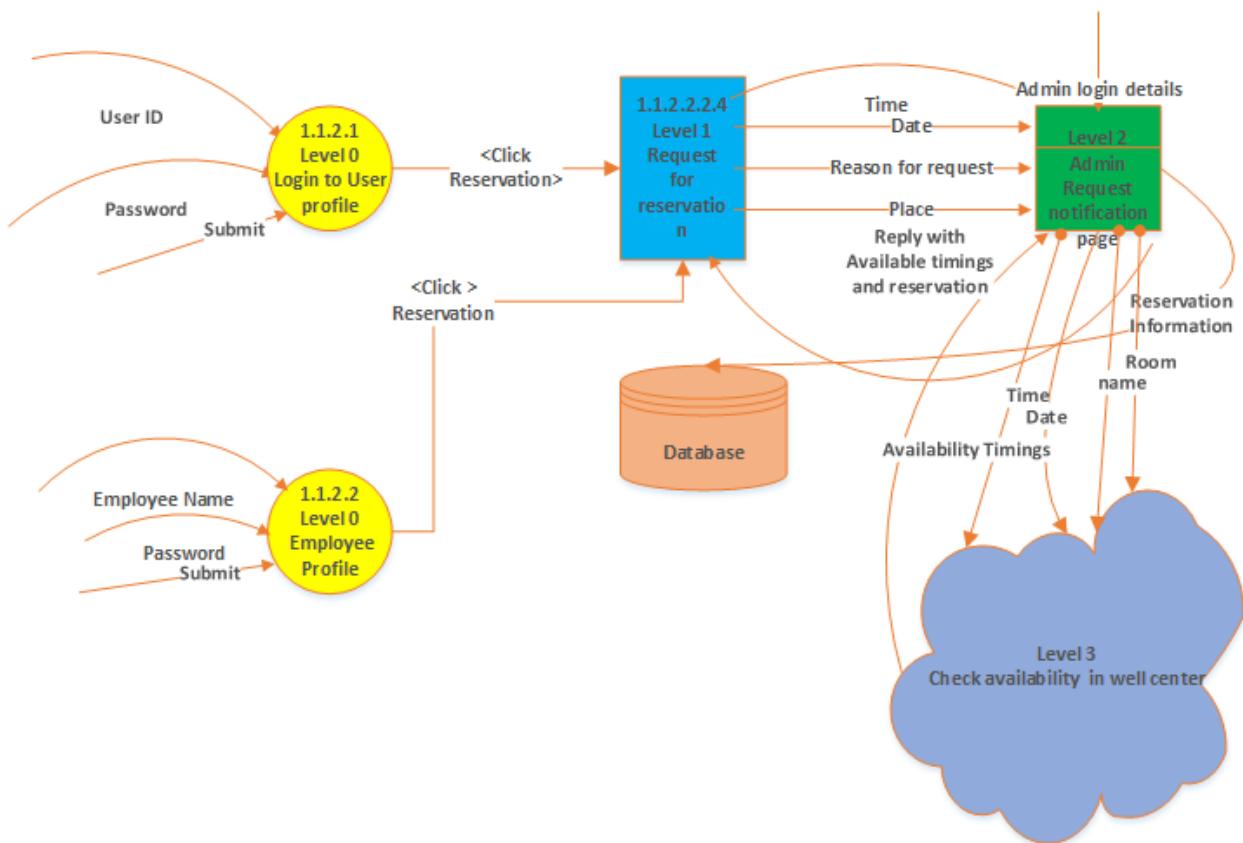
Screenshot of suspension process (WBS NO 1.1.2.3.2)



### 1.1.2.3.3 Reservation

WBS No. 1.1.2.3.3 Reservation	
Description	The reservation system will be accessed by the administrator to book the system with the place reason, time and the date for organizing meeting in the wellness center
Sprint	S18
Developer team	Hussain, Appala, Shaohu
Output	The reservation will be done at the timings available in the SDSU wellness center.

Data flow diagram for Reservation System:



## Screenshot of Reservation

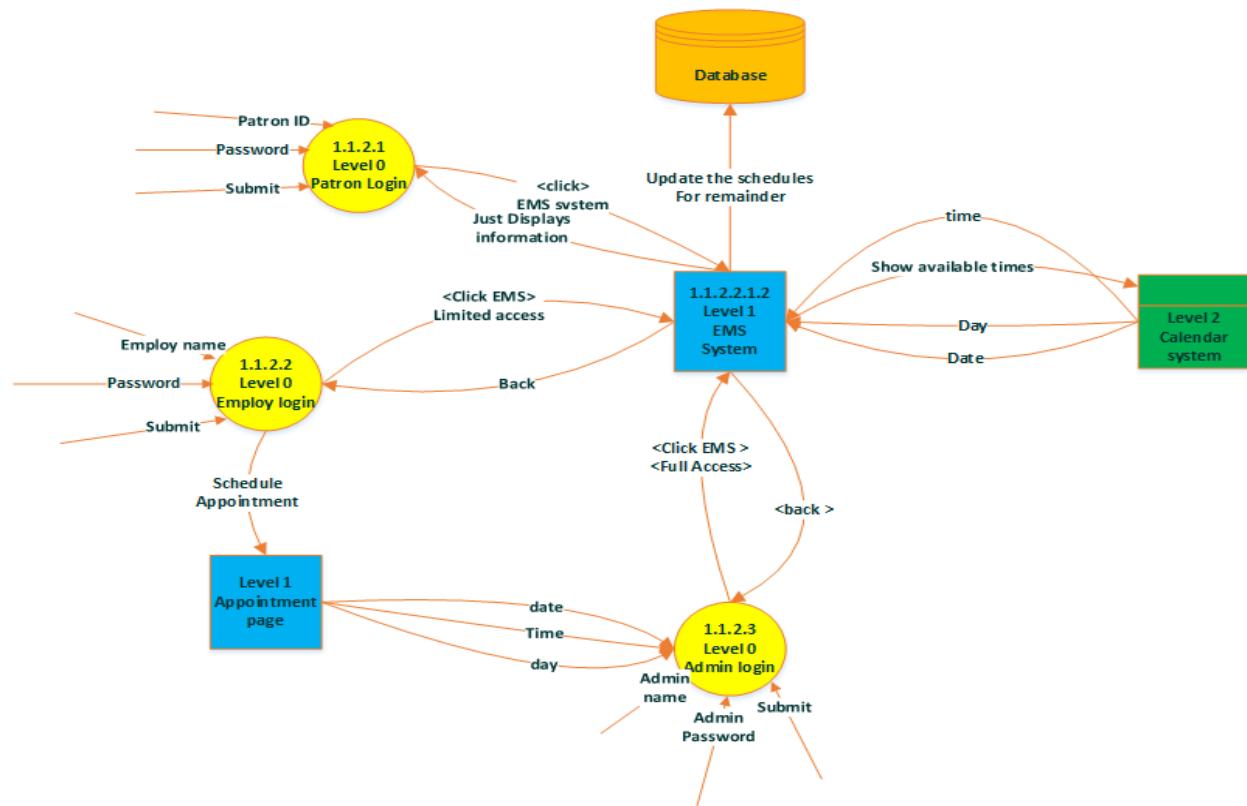
### Details

Add a title for the event	
Add a location	
Add room	
Start	
Mon 10/24/2016	9:00 AM
<input type="checkbox"/> All day	
End	
Mon 10/24/2016	9:30 AM
<input type="checkbox"/> Private	
Repeat	
Never	Save to calendar
Calendar	
Reminder	
15 minutes	Show as
Busy	
<a href="#">Add an email reminder</a>	

#### 1.1.2.3.4 EMS system

WBS No. 1.1.2.3.4 EMS System	
Description	Display Scheduling System for appointment, Class Timings and the notification remainder and soon. The admin will have the full access to the EMS system.
Sprint	S19
Developer team	Hussain, Appala, Shaohu
Output	The available schedules will be the limited access available timings for the employee for the class timings, notification reminders and that will be displayed on the patron page.

Data flow diagram for the EMS system (WBS NO 1.1.2.3.4)



Screenshot of EMS system (WBS NO 1.1.2.3.4)

## Details

Add a title for the event

Add a location

Start  
Mon 10/24/2016  9:00 AM   All day

End  
Mon 10/24/2016  9:30 AM   Private

Repeat  Save to calendar

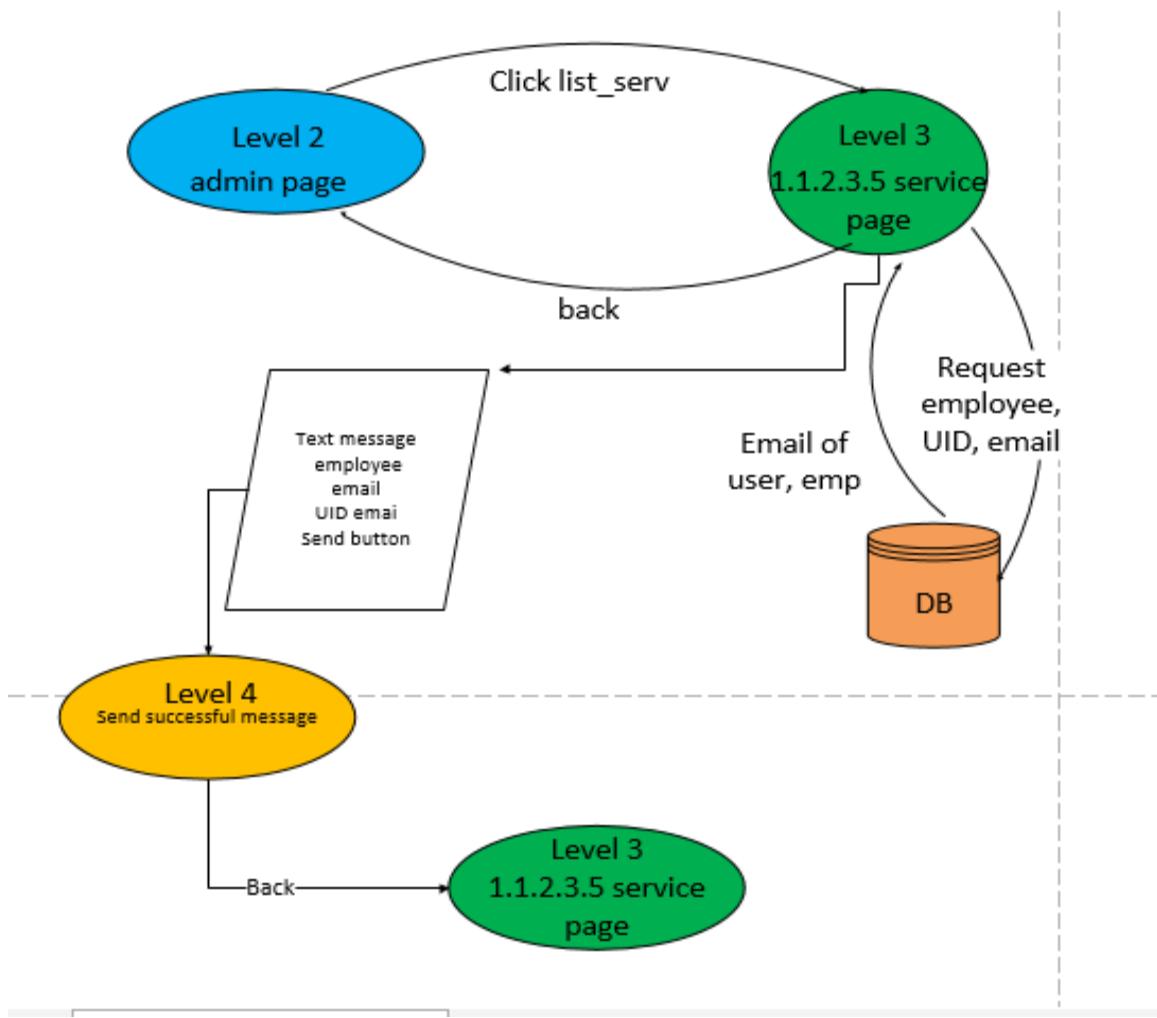
Reminder  Show as

[Add an email reminder](#)

### 1.1.2.3.5 List Serv

WBS No. 1.1.2.3.6 List Serv	
Description	This function will let admin to get contact with patrons
Sprint	S20
Developer team	Hussain, Appala, Shaohu
Output	Generate a list with all emails address for patrons and employees

Data flow for List Serv(WBS NO 1.1.2.3.5)



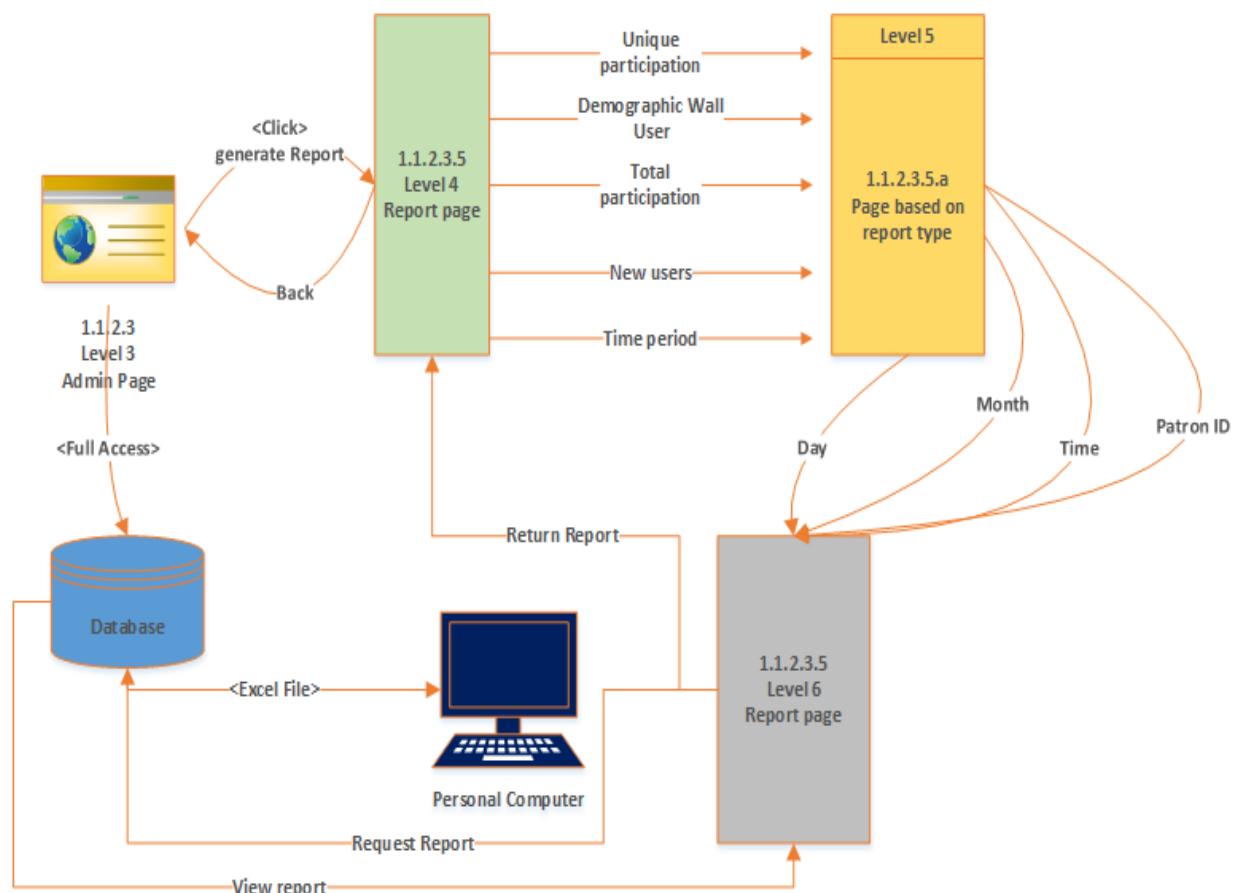
Screenshot of List Serv (WBS NO 1.1.2.3.5)

A	B	C	D	E	F	G	H	I	J
Patron report with total participation									
S-NO	First_name	Last_name	Email	phone	expiration date	attended	from_date	End_date	
1	Hussain	Otudi	Ot@jacks	605-592	12/16/16	40	10/10/2016	10/16/2016	
2	shaoh	zhang	Sh@jacks	605-444	12/17/16	35	10/10/2016	10/16/2016	
3	Lee	haney	Lee@jacks	605-232	12/18/16	40	10/10/2016	10/16/2016	
4	Shin	sung	Shin@jacks	605-232	12/19/16	32	10/10/2016	10/16/2016	
5	Rajo	appala	Rajo@jacks	605-232	12/20/16	27	10/10/2016	10/16/2016	
6	Sayun	sahu	Sayun@jacks	605-232	12/21/16	40	10/10/2016	10/16/2016	
7	Bader	alzeidi	Bader@jacks	605-232	12/22/16	25	10/10/2016	10/16/2016	
8	Ali	selhenia	Ali@jacks	605-232	12/23/16	20	10/10/2016	10/16/2016	
9	Min	manki	Min@jacks	605-232	12/24/16	15	10/10/2016	10/16/2016	
10	Hamer	george	Hamer@jacks	605-232	12/25/16	40	10/10/2016	10/16/2016	
11									
12									
13									
14									
15									

### 1.1.2.3.6 report generation

WBS No. 1.1.2.3.6 Report Generation	
Description	The report management will provide the report of the wall climbers based on the admin requirement.
Sprint	S21
Developer team	Hussain, Appala, Shaohu
Output	The return report will be stored on the system and will be displayed on the screen of the admin.

Data Flow diagram for Report generation (WBS NO 1.1.2.3.6)



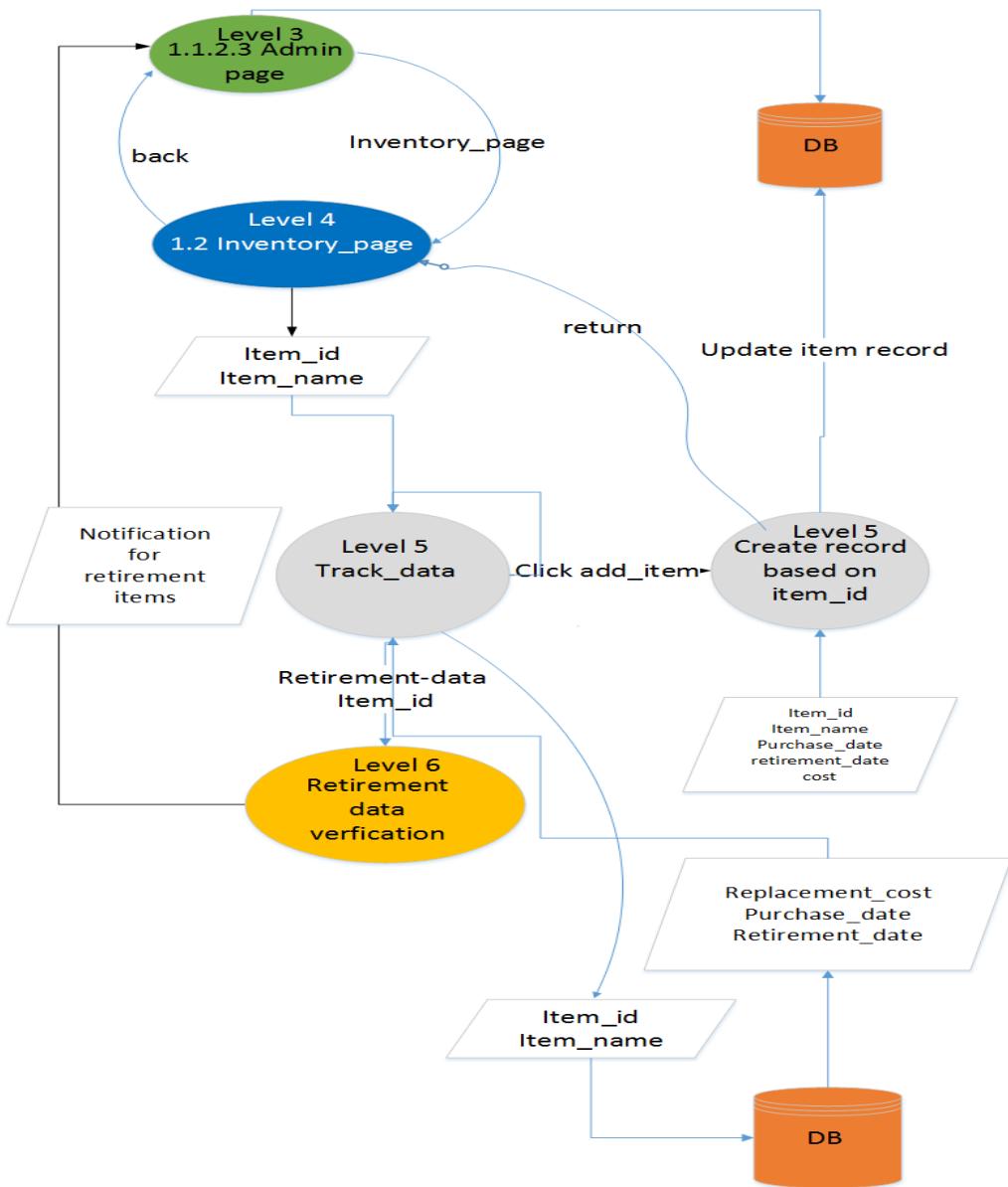
Screenshot of report generation (WBS NO 1.1.2.3.6)

	A	B	C	D	E	F	G	H	I	J
1	Patron report with unique participation									
2	S-NO	First_name	Last_name	Email	phone	expiration date	attended	from_date	End_date	
3	1	Hussain	Otudi	Ot@jacks	605-592	12/16/16	40	10/10/2016	10/16/2016	
4	2	shaoh	zhang	Sh@jacks	605-444	12/17/16	35	10/10/2016	10/16/2016	
5	3	Lee	haney	Lee@jacks	605-232	12/18/16	40	10/10/2016	10/16/2016	
6	4	Shin	sung	Shin@jacks	605-232	12/19/16	32	10/10/2016	10/16/2016	
7	5	Rajo	appala	Rajo@jacks	605-232	12/20/16	27	10/10/2016	10/16/2016	
8	6	Sayun	sahu	Sayun@jacks	605-232	12/21/16	40	10/10/2016	10/16/2016	
9	7	Bader	alzeidi	Bader@jacks	605-232	12/22/16	25	10/10/2016	10/16/2016	
10	8	Ali	selhenia	Ali@jacks	605-232	12/23/16	20	10/10/2016	10/16/2016	
11	9	Min	manki	Min@jacks	605-232	12/24/16	15	10/10/2016	10/16/2016	
12	10	Hamer	george	Hamer@jacks	605-232	12/25/16	40	10/10/2016	10/16/2016	
13										
14										

## 1.2 Inventory

WBS No. 1.2 inventory	
Description	This function will allow admin to track item based on id and purchase date, retirement date and replacement cost
Sprint	S22
Developer team	Hussain, Appala, Shaohu
Output	The output will be a notification that inform admin about item when is getting close to retirement

Data flow diagram for inventory (WBS NO 1.2)



Screenshot of inventory (WBS NO 1.2)

https://www.sdsstate.edu/wellness-center/climbing-wall/Inventory/homepage.php

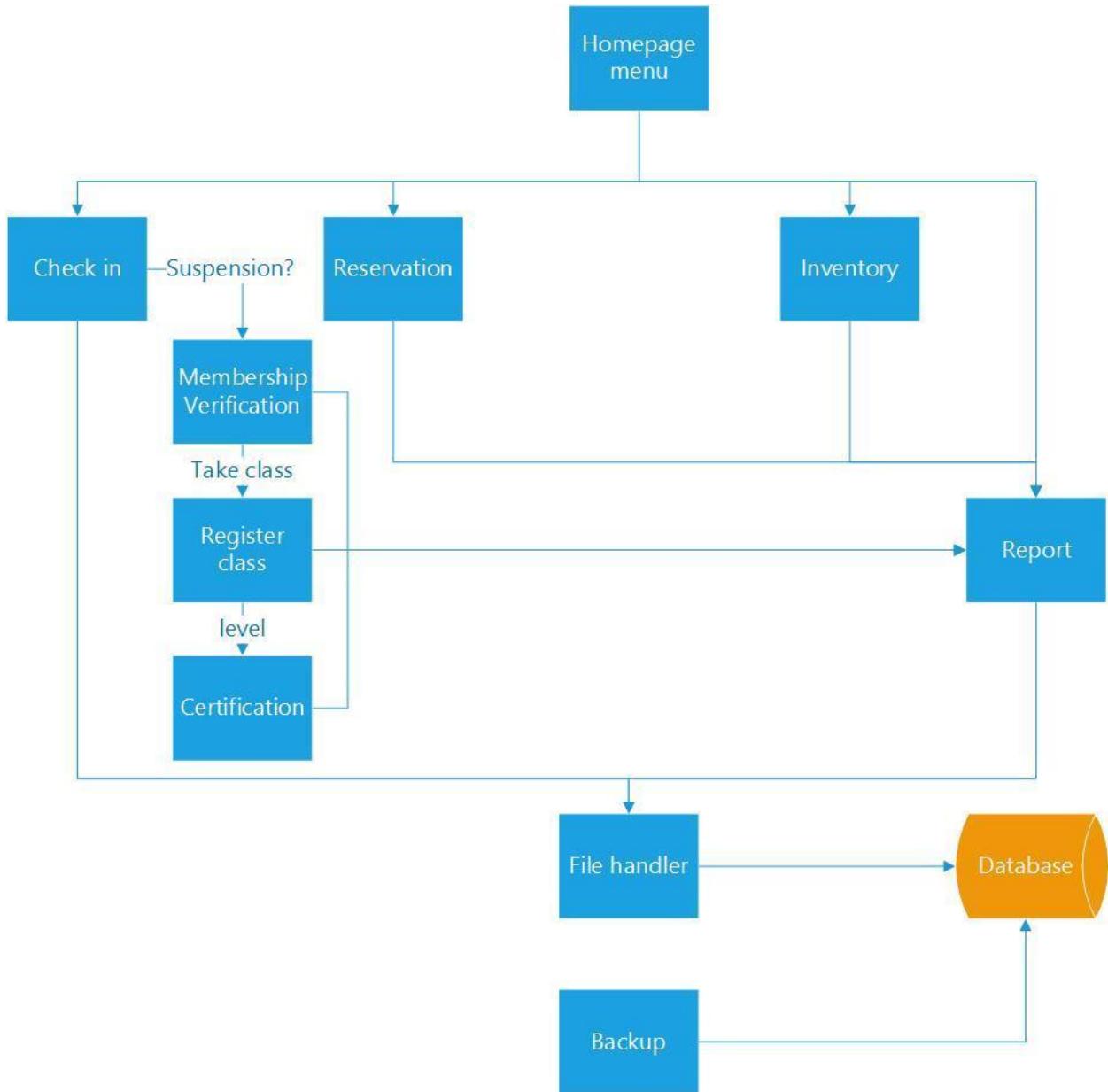
**Wall Climbing Center**

Log in  
sign up

Home	Inventory	POS	Contact Us
Pos System	Equipment Information	Binding Page	
Start and remove information	No Item Start date End date Price		
Eqipment List	1 Rope Climbing Harness 10/9/2016 11/9/2016 \$99		
Update information	2 Climbing Harness 10/9/2016 11/9/2016 \$299		
Notification	3 Chest Harness 10/9/2016 11/9/2016 \$145		
	4 Climbing Shoes 10/9/2016 11/9/2016 \$199		

South Dakota State University  
Brookings, SD 57007  
1.800.952.3541  
Copyright © 2016

## 5. TOP LEVEL DESIGN



### 5.1 Performance constraints

**Table 10. Performance constraints**

Items	Description
Processor cycles	The wait time at higher usages for access to the application can be

	significant if the processor cycle is long .
Storage	The history data need a certain amount of real storage.
Database-associated hardware (I/O)contention	I/O operations passes through many devices if these devices overused the time taken to access the data can increase.
User interface	Good user interface leads the project running smoothly.

## 5.2 Fault handling approach

Table1 11. Fault handling approach		
Items	Examples	Approaches
Software Fault Handling	<ul style="list-style-type: none"> <li>• faults in 3rd party software libraries</li> <li>• software faults and bugs in packaged applications</li> </ul>	<ul style="list-style-type: none"> <li>• Set up a try/catch block</li> <li>• Use of try/finally blocks around code that can potentially generate an exception and centralize the catch statements in one location</li> <li>• Ordering exceptions in catch blocks from the most specific to the least specific.</li> <li>• End exception class names with the word "Exception".</li> <li>• Return null for extremely common error cases instead of throwing an exception.</li> </ul>
Hardware Fault Handling	<ul style="list-style-type: none"> <li>• Fails of mechanical parts, such as fans and disk drives</li> </ul>	<ul style="list-style-type: none"> <li>• Drivers and framework update/installation</li> <li>• Bandwidth configuration</li> <li>• Hardware security</li> <li>• Firewalls</li> </ul>
Technical errors	<ul style="list-style-type: none"> <li>• Faults caused by errors in the underlying infrastructure or middleware components such as network errors, server failures, corrupt disks and so on.</li> </ul>	<ul style="list-style-type: none"> <li>• Frequently maintain facility</li> </ul>

## 6. MEDIUM LEVEL DESIGN

This section gives more detail about requirement functionality.

### 1.1.1 Registration of patron

WBS No. 1.1.1 registration	
Description	Allows user to enter the information and validate the information.
Validation	Valid email address
Input	First_name, last_name, email, confirm_email, password, confirm password , phone, DOB, Sex, and submit action
Output	Successful message with user unique id and his information.
Flow Chart	<pre> graph LR     Start((Registration System)) --&gt; Username([Username])     Username --&gt; FirstName([First name])     FirstName --&gt; LastName([Lastname])     LastName --&gt; Age([Age])     Age --&gt; Sex([Sex])     Sex --&gt; Password([Password])     Password --&gt; ConfirmPassword([Confirm Password])     ConfirmPassword --&gt; EmailAddress([Email Address])     EmailAddress --&gt; PhoneNumber([Phone number])     PhoneNumber --&gt; Confirm([Confirm])     Confirm --&gt; VerifyEmail{Verify email In database}     VerifyEmail -- Already registered --&gt; LogIn([Log In])     VerifyEmail -- If not --&gt; RandomIdAlgorithm[/Random Id number Generation Algorithm/]     RandomIdAlgorithm --&gt; CreateAccount[Create account]     CreateAccount -- Enter --&gt; PatronLogin[Patron Login]     CreateAccount --&gt; UpdateDatabase[Updated Database]     UpdateDatabase -- Update to database --&gt; CreateAccount   </pre> <p>The flowchart illustrates the registration process. It starts with the 'Registration System' which triggers the sequence of input fields: Username, First name, Lastname, Age, and Sex. These are followed by Password, Confirm Password, Email Address, and Phone number. A 'Sign In' step leads to the 'Confirm' step. From 'Confirm', the process branches: if the email is already registered, it leads to 'Log In'; if not, it triggers a 'Random Id number Generation Algorithm' which then leads to 'Create account'. 'Create account' is connected to 'Patron Login' via an 'Enter' action. Finally, 'Create account' leads to 'Updated Database' via an 'Update to database' action.</p>
Pseudocode	<pre> import java.io.*; import java.sql.*; import javax.servlet.ServletException;   </pre>

```

import javax.servlet.http.*;

public class Register extends HttpServlet {
    public void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        String n=request.getParameter("userName");
        String p=request.getParameter("userPass");
        String p=request.getParameter("password");
        String p=request.getParameter("confirm password");
        String p=request.getParameter("email");
        String p=request.getParameter("confirm email");
        String p=request.getParameter("phone number");
        String p=request.getParameter("age");
        String p=request.getParameter("date of birth");
        String e=request.getParameter("Submit");
        String c=request.getParameter("Reset");

        try{
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con=DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521:xe","system","oracle");

            PreparedStatement ps=con.prepareStatement(
                "insert into registeruser values(?,?,?,?,?,?,?,?,?,?,?,?)");

            ps.setString(1,n);
            ps.setString(2,p);
            ps.setString(3,e);
            ps.setString(4,c);
            ps.setString(5,c);
            ps.setString(6,c);
            ps.setString(7,c);
            ps.setString(8,c);
            ps.setString(9,c);
            ps.setString(10,c);
            ps.setString(11,c);

            int i=ps.executeUpdate();
            if(i>0)
                out.print("You are successfully registered...");

        }catch (Exception e2) {System.out.println(e2);}

        out.close();
    }
}

```

	}
Screen shot	

### 1.1.1.1 waiver form

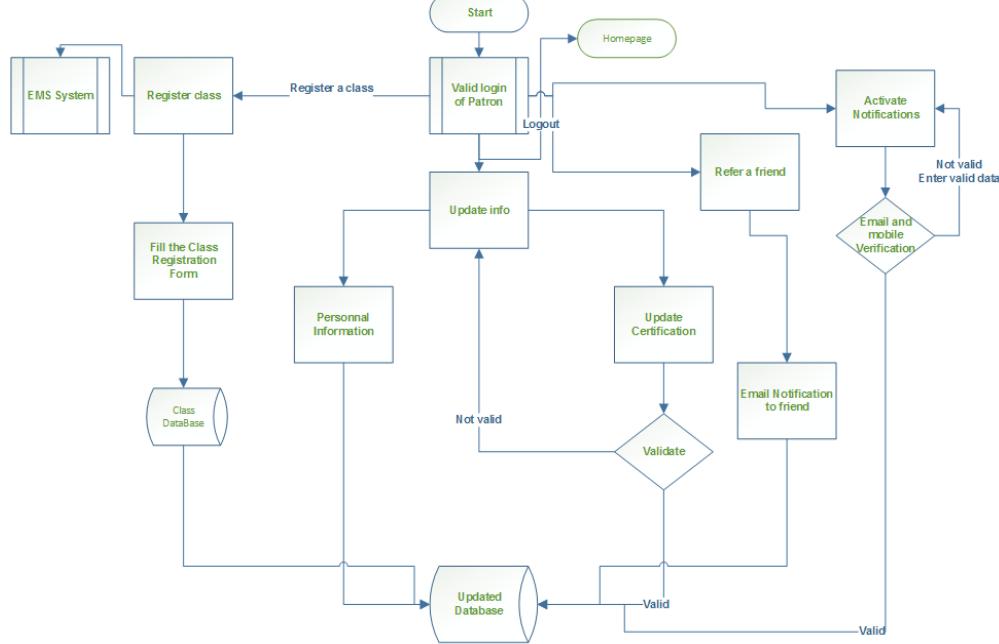
WBS No. 1.1.1.1 waiver form	
Description	The waiver form will give the description of the risk management policies of the rock climbing center
Validation	If the waiver form is submitted or not
Input	Patron name ,ID, with the acceptance of the waiver form by the patron
Output	The waiver form of the patron will be updated to the database in the form of the pdf file
Pseudocode	<pre> jButton1 = new JButton(); getContentPane().add(jButton1); jButton1.setText("Waiver form"); jButton1.setText("Terms and policies"); jButton1.setText("AcceptReject"); jButton1.addActionListener(new ActionListener() {     public void actionPerformed(ActionEvent evt) {         System.out.println("jButton1.actionPerformed, event=" + evt);         AcceptReject ask = new AcceptReject(); //&lt;----- Red console errors flag this line         ask.setVisible(true);     } }); </pre>

Screen shot	 <p><b>SDSU Climbing Gym Acknowledgement of Risk</b></p> <p><b>PLEASE NOTE: This Waiver of Liability, Release, Acknowledgement of Risk, and Indemnification Agreement ("Waiver Agreement") is intended to be, and is, legally binding.</b></p> <p>If any aspect of this Waiver Agreement requires clarification, have an SDSU Wellness Center Climbing Gym attendant fully explain it, before signing. By signing the Wellness Center Climbing Gym "Sign-in Sheet(s)", you are agreeing to all terms set forth in this Waiver Agreement. You and/or the person on whose behalf you are signing, are waiving the right to bring any type of action, whether in court or otherwise, to recover compensation or obtain any other remedy for any personal injuries, damages to property, any accident or incident of any type, or death, arising out of or related to your use of the Wellness Center Climbing Gym, its facilities, grounds, climbing walls, exercise areas, equipment, whether the use is supervised or unsupervised. Rock climbing is a sport that has inherent risks. While the SDSU Wellness Center Climbing Gym offers the sport of rock climbing in a controlled environment, there is still an assumed risk of injury to persons using the SDSU Wellness Center Climbing Gym. In agreeing to this Waiver Agreement, I hereby acknowledge, understand, and agree on my behalf, and upon behalf of the person for whom I am signing, that the sport of rock climbing and the use of the SDSU Wellness Center Climbing Gym has inherent risks. These risks include, but are not limited to any injury of damage resulting from:</p> <p>Negligence of employees, or the SDSU Wellness Center Climbing Gym. Negligent misuse of the facility, climbing walls, or equipment of the SDSU Wellness Center Climbing Gym; Falling off or impacting against the climbing walls, impact surface, floors, or anything else; Rope abrasion, entanglement or other activities occurring on the premises; Cuts or abrasions resulting from any cause whatsoever; Failure of the climbing walls or equipment, whether inside or outside; Personal health problems, whether mental or physical; Negligence of other climbers, visitors, or observers or persons who may be present in or around the climbing area or facility; and/or Negligence or lack of adequate training or any person(s) who seek to assist with medical or other help either before or after any injury or damage may occur.</p> <p>By signing the "Sign-in Sheet", I, for myself and for my heirs, next of kin assigns, and personal representatives, hereby agree to and do release, indemnify and hold harmless the State of South Dakota, South Dakota Board of Regents, South Dakota State University, and the SDSU Wellness Center Climbing Gym, and their officers, employees, and agents, and/or volunteer assistants, from any and all injuries and damage which I, or the person upon whose behalf I am signing in, may sustain or incur arising out of or related to my use of the SDSU Wellness Center Climbing Gym, its facilities, grounds, climbing walls, exercise areas, equipment, participation in classes or events, and/or outdoor programs guided by or connected with the SDSU Wellness Center Climbing Gym, whether the use is supervised or unsupervised. I, for myself and for my heirs, next of kin, assigns, personal representatives, and persons upon whose behalf I am signing the "Sign-in Sheet," hereby agree to and release, indemnify and hold harmless the State of South Dakota, South Dakota Board of Regents, South Dakota State University, and the SDSU Wellness Center Climbing Gym, and their agents, employees, offices, and, volunteer assistants, from any and all causes of action, claims for damages or demands whatsoever. THIS WAIVER AGREEMENT IS BINDING EVEN IF THE RELEASED PERSON(S) OR ENTITY(IES) HAVE CAUSED OR CONTRIBUTED TO ANY DAMAGE OR INJURY THROUGH THEIR COLLECTIVE OR INDIVIDUAL NEGLIGENCE.</p> <p><i>I and/or person on whose behalf I am signing-in, voluntarily assume complete responsibility for risks and any injuries or damage which may occur as a result of those risks even if the manner or type of injury or damage occurs in a manner that is not foreseeable at the time this Waiver Agreement is accepted. In consideration of my use of the gym, its equipment, employees, volunteer assistants, independent contractors, I agree to and do release, indemnify and hold harmless, the SDSU Wellness Center Climbing Gym, and any and all of their agents, servants and employees, from all liability, claims, demands and damages and further promise not to commence any action or proceeding asserting same.</i></p> <p>All climbers who are twelve (12) years of age or under must be directly supervised by an SDSU Wellness Center Climbing Gym approved adult or be a participant in an SDSU Wellness Center Climbing Gym program. The "Sign-in Sheet" is being signed by the youth's parent, legal guardian, or adult authorized to sign by the youth's parent or legal guardian. By signing the "Sign-in Sheet", the adult acknowledges that they understand the terms of the Waiver Agreement and has the authority to sign for the youth climber. The person signing the "Sign-in Sheet" understands and acknowledges that this Waiver Agreement is binding on the person on whose behalf the "Sign-in Sheet" is signed, for their heirs, next of kin, assigns, and personal representatives.</p> <p>BY SIGNING THE SDSU Wellness Center Climbing Gym "SIGN-IN SHEET" I ACKNOWLEDGE THAT I HAVE READ AND AGREE TO THE TERMS OF THIS WAIVER AGREEMENT. THERE ARE NO ORAL REPRESENTATIONS, STATEMENTS, OR INDUCEMENTS WHICH HAVE BEEN MADE THAT ALTER, CHANGE OR MODIFY ANYTHING SET FORTH IN THIS WAIVER AGREEMENT.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center; width: 10%;"> </th><th style="text-align: center; width: 20%;">Print Name</th><th style="text-align: center; width: 20%;">Sign Name</th><th style="text-align: center; width: 10%;">Member Type</th><th style="text-align: center; width: 10%;">Gender: *Optional*</th><th style="text-align: center; width: 15%;">Date &amp; Time Arrived</th><th style="text-align: center; width: 15%;">Time Left</th></tr> </thead> <tbody> <tr> <td style="text-align: center;">1.</td><td></td><td></td><td></td><td></td><td></td><td></td></tr> </tbody> </table>		Print Name	Sign Name	Member Type	Gender: *Optional*	Date & Time Arrived	Time Left	1.						
	Print Name	Sign Name	Member Type	Gender: *Optional*	Date & Time Arrived	Time Left									
1.															

### 1.1.1.2 user validation

WBS No. 1.1.1.2 user validation	
Description	Here the Email of the user will get verified in the database in order to remove the duplication creation of the patron account.
Validation	If the email exists or not
Input	The patron email address will be given as the input for the algorithm.
Output	The email will be verified with the database for the patron before creating the database for the new registered patron.

Flow Chart



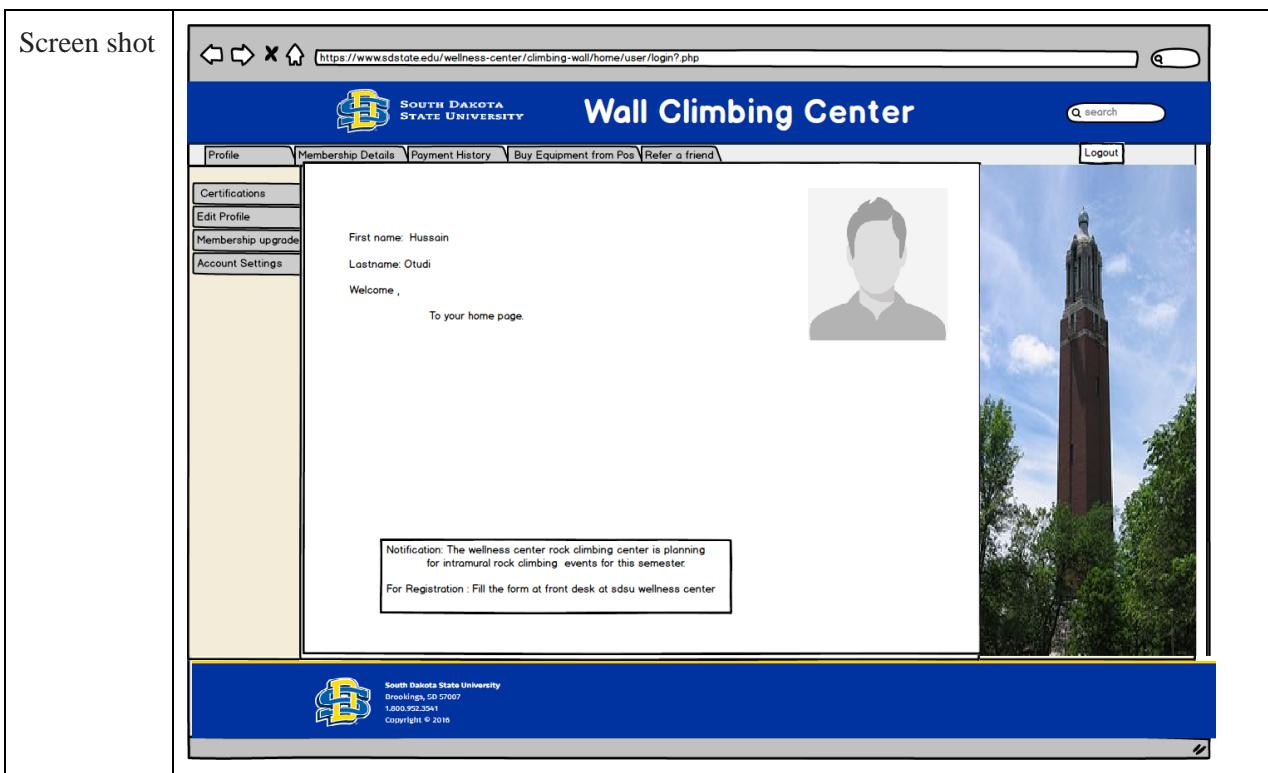
Pseudocode

```

public class Login {
public void run() {
    Scanner scan = new Scanner (new File("the\\dir\\myfile.extension"));
    Scanner keyboard = new Scanner (System.in);
    String user = scan.nextLine();
    String pass = scan.nextLine(); // looks at selected file in scan

    String inpUser = keyboard.nextLine();
    String inpPass = keyboard.nextLine(); // gets input from user

    if (inpUser.equals(user) && inpPass.equals(pass)) {
        System.out.print("your login message");
    } else {
        System.out.print("your error message");
    }
}
}
  
```



#### 1.1.1.2.d) Random ID Generation

WBS No. 1.1.1.4 update user information	
Description	The user once created an account then the Random ID will get generated and stored in the database.
Input	Click submit button for the to enter user.
Output	The user data is stored based on the unique ID created.
Pseudocode	<pre> static int counter = 0; public static String get_new_id(){     String random_id = (String(counter) + rand_seed);     counter = counter + 1;     return random_id; } </pre>

### 1.1.1.3 payment method

WBS No. 1.1.1.3 Payment method	
Description	Here the payment method can be done for the Membership of the patron from his profile and also for the POS system.
Input	The input will be card details of the credit card\debit card and the cash payment at front desk based on the patron ID
Output	The output will be the payment successful/unsuccessful message shown based on the process verification and updated with the Patron ID.
Flowchart	<pre>     graph TD         START([START]) --&gt; PP[Patron Payment method]         PP --&gt; PT{Payment Type}                  PT --&gt; CP[Cash Payment]         PT --&gt; CDB[Credit/Debit Card]                  CP --&gt; EBD[Enter bank details]         EBD --&gt; VPD{Validate Payment}                  CDB --&gt; ECD[Enter Card Details]         ECD --&gt; RP{Remember Payment details}         RP -- yes --&gt; UDB(Update Database)         RP -- no --&gt; PD[Payment Done]                  VPD -- Valid --&gt; PD         VPD -- Not valid --&gt; EBD         VPD -- Enter valid Bank details --&gt; EBD         VPD -- Enter correct details of your card --&gt; ECD     </pre> <p>The flowchart starts with a 'START' node, leading to a process box 'Patron Payment method'. This leads to a decision diamond 'Payment Type'. From 'Payment Type', two paths emerge: 'Cash Payment' and 'Credit/Debit Card'. The 'Cash Payment' path leads to 'Enter bank details', which then leads to a decision diamond 'Validate Payment'. The 'Credit/Debit Card' path leads to 'Enter Card Details', then to 'Remember Payment details'. From 'Remember Payment details', a 'yes' branch leads to an update database node 'Update Database', and a 'no' branch leads to 'Payment Done'. The 'Validate Payment' diamond has three outcomes: 'Valid' leads to 'Payment Done'; 'Not valid' leads back to 'Enter bank details'; and 'Enter valid Bank details' or 'Enter correct details of your card' both lead back to 'Enter bank details'.</p>
Pseudocode	<pre> &lt;form id="simplify-payment-form" action="" method="POST"&gt;     &lt;div&gt;         &lt;label&gt;Credit Card Number: &lt;/label&gt;         &lt;input id="cc-number" type="text" maxlength="20" autocomplete="off" value="" autofocus /&gt;     &lt;/div&gt;     &lt;div&gt;         &lt;label&gt;CVC: &lt;/label&gt;         &lt;input id="cc-cvc" type="text" maxlength="4" autocomplete="off" /&gt;     &lt;/div&gt; </pre>

```

        value="" />
    </div>
    <div>
        <label>Expiry Date: </label>
        <select id="cc-exp-month">
            <option value="01">Jan</option>
            <option value="02">Feb</option>
            <option value="03">Mar</option>
            <option value="04">Apr</option>
            <option value="05">May</option>
            <option value="06">Jun</option>
            <option value="07">Jul</option>
            <option value="08">Aug</option>
            <option value="09">Sep</option>
            <option value="10">Oct</option>
            <option value="11">Nov</option>
            <option value="12">Dec</option>
        </select>
        <select id="cc-exp-year">
            <option value="13">2013</option>
            <option value="14">2014</option>
            <option value="15">2015</option>
            <option value="16">2016</option>
            <option value="17">2017</option>
            <option value="18">2018</option>
            <option value="19">2019</option>
            <option value="20">2020</option>
            <option value="21">2021</option>
            <option value="22">2022</option>
        </select>
    </div>
    <button id="process-payment-btn" type="submit">Process Payment</button>
</form>

```

Screen shot

**Wall Climbing Center**

Pay with a debit or credit card  
If you don't have a PayPal account

Country

Credit Card number

VISA

Expiration Date  /  CSC  [What's this?](#)

First name

Last name

Billing Address Line 1

Billing Address Line 2 (Optional)

Zip code

City

State

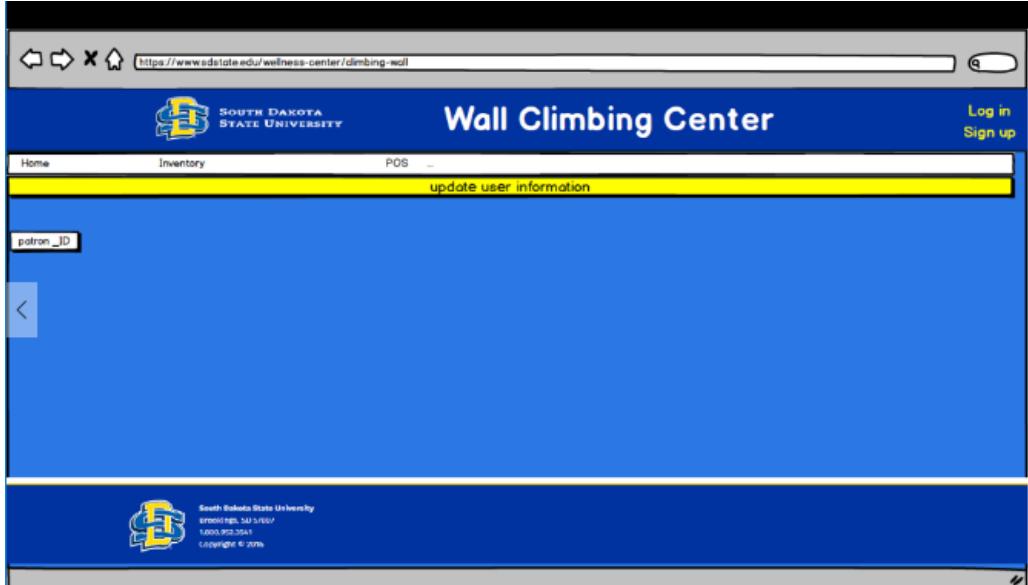
Phone e.g. 555-555-5555

Email Address

[Make Payment](#)

South Dakota State University  
Brookings, SD 57007  
1.800.952.3341  
Copyright © 2016

#### 1.1.1.4 Update user information

WBS No. 1.1.1.4 update user information	
Description	This function let user update his/her contact information.
Input	Email, phone_number, and address
Output	The output will display successful message that his information has been updated
Screen shot	 <p>A screenshot of a web browser window. The title bar says "WBS No. 1.1.1.4 update user information". The main content area shows a blue-themed webpage for the "Wall Climbing Center" at South Dakota State University. The page includes a navigation bar with links for Home, Inventory, POS, Log in, and Sign up. A yellow banner across the middle says "update user information". Below the banner, there is a form field labeled "patron_ID". At the bottom of the page, there is a footer with the university's logo and some copyright information.</p>

#### 1.1.2 Log In

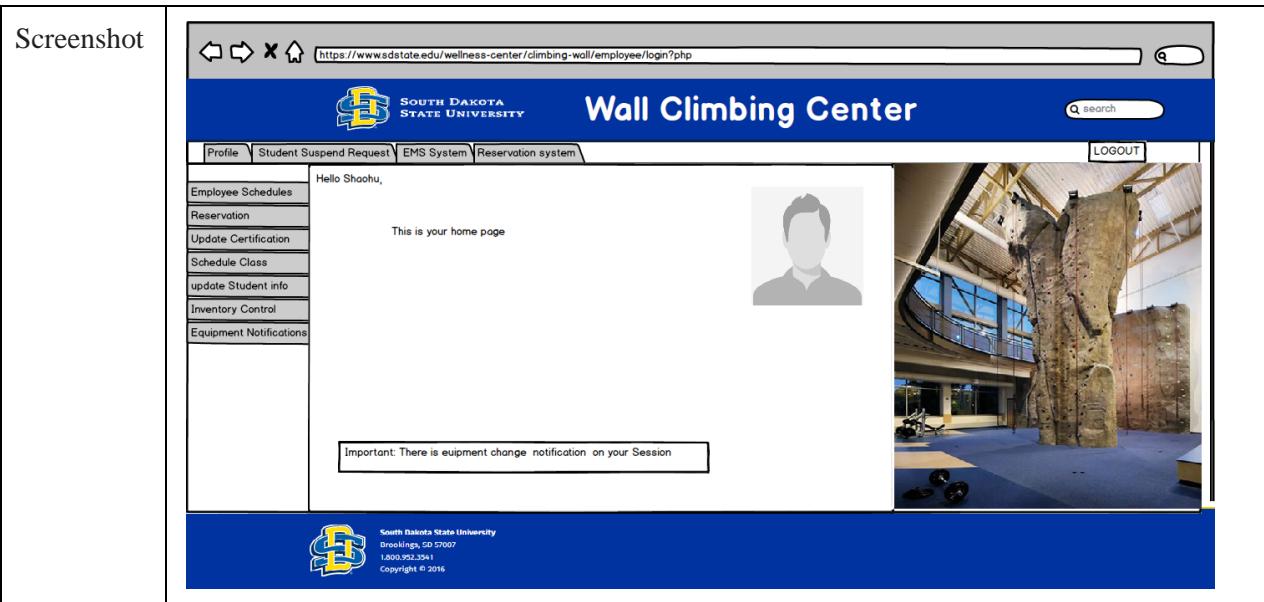
WBS No. 1.1.2 login	
Description	This function for parton login, it will be displayed as html page asking user to enter username and password
Input	Here the input will be user_id and password
Output	The page will display for user include his/ her name on the top of homepage

Flow chart	<pre> graph TD     Start([Start]) --&gt; LoginPage[Login page]     LoginPage -- Not valid --&gt; Homepage((Homepage))     LoginPage -- valid --&gt; AfterLogin{After login}     AfterLogin --&gt; Staff[Staff]     AfterLogin --&gt; Instructor[Instructor]     Staff --&gt; RegisterStudents[Register Students to class]     Instructor --&gt; RegisterStudents     RegisterStudents --&gt; VerifyPayment{Verify Students Payment}     VerifyPayment -- Payment done --&gt; ScheduleClass[Schedule class timings for students]     VerifyPayment -- Payment not done --&gt; PaymentProcess[Payment Process]     PaymentProcess --&gt; VerifyPayment     ScheduleClass --&gt; EMS[EMS]     EMS --&gt; UpdateAttendance[Update Attendance Of class]     UpdateAttendance --&gt; SendNotification[Send Notification About schedule and attendance]     VerifyPayment --&gt; ValidateResults{Validate Results after Certifications}     ValidateResults -- Passed the test --&gt; AllocateCertification[Allocate Certification]     AllocateCertification -- Update Certification --&gt; UpdatedDatabase((Updated DataBase))     UpdatedDatabase --&gt; VerifyPayment     VerifyPayment --&gt; Staff     VerifyPayment --&gt; Instructor     Staff --&gt; VerifyPayment     ValidateResults -- Not passed --&gt; VerifyPayment     AllocateCertification --&gt; VerifyPayment     </pre> <p>1.1.2.2.1 EMS 1.1.2.2.2 Check in 1.1.2.2.3 Send notifications 1.1.2.2.4 Reservation 1.1.2.2.5 Request Patron Suspension 1.1.2.2.6 update Certification 1.1.2.2.7 Daily notes</p>
pseudocode	<pre> public class Login { public void run() {     Scanner scan = new Scanner (new File("the\\dir\\myfile.extension"));     Scanner keyboard = new Scanner (System.in);     String user = scan.nextLine();     String pass = scan.nextLine(); // looks at selected file in scan      String inpUser = keyboard.nextLine();     String inpPass = keyboard.nextLine(); // gets input from user      if (inpUser.equals(user) &amp;&amp; inpPass.equals(pass)) {         System.out.print("your login message");     } else {         System.out.print("your error message");     } } } </pre>



### 1.1.2.2 employee

WBS No. 1.1.2.2 employee	
Description	This function for employee login, will be displayed as in html page asking employee to enter his/her name and pwd
Input	The input will be the name of employee and password ,
Output	The employee page will be displayed with its details



#### 1.1.2.2.1.1 Register class

WBS No. 1.1.2.2.1.1 Registering student to a class for certification	
Description	Here the employ/ staff will register the student into the class for certification
Input	The inputs will be student name, Patron ID, Email, phone number, Instructor name, and the user record created.
Output	The patron with all his/her details will be created a record and then updated it to class list.

Flow chart	<pre> graph TD     Start((Start)) --&gt; LoginPage[Login page]     LoginPage -- "Not valid" --&gt; Homepage((Homepage))     LoginPage -- "valid" --&gt; AfterLogin([After login])     AfterLogin --&gt; Staff[Staff]     AfterLogin --&gt; Instructor[Instructor]     Staff --&gt; RegisterStudents[Register Students to class]     Instructor --&gt; RegisterStudents     RegisterStudents --&gt; VerifyPayment{Verify Students Payment}     VerifyPayment -- "Payment not done" --&gt; PaymentProcess[Payment Process]     VerifyPayment -- "Payment done" --&gt; ScheduleClassTimings[Schedule class timings for students]     ScheduleClassTimings --&gt; UpdateAttendance[Update Attendance Of class]     ScheduleClassTimings --&gt; SendNotification[Send Notification About schedule and attendance]     UpdateAttendance --&gt; UpdatedDatabase((Updated Database))     SendNotification --&gt; UpdatedDatabase     AfterLogin -- "Request to register If failed" --&gt; ValidateResults{Validate Results after Certifications}     ValidateResults -- "Passed the test" --&gt; AllocateCertification[Allocate Certification]     AllocateCertification -- "Update Certification" --&gt; UpdatedDatabase     </pre>
Pseudocode	<pre> public class Patron {     private String first_name;     private String last_name;     private String email;     private int patron_id;     private int phone_number;     private String class_name;     private String instructor_name;      // construct a new Patron with given fields     public Patron(String first_name, String last_name, String email, int patron_id,int phone_number,String class_name,instructor_name ) {         this.first_name = first_name;         this.last_name = last_name;         this.email = email;         this.patron_id = patron_id;         this.phone_number = phone_number;         this.class_name= class_name;     } } </pre>

```

        this.instructor_name= instructor_name;
    }

// return true if the invoking object's patron_id is less than that of b
public boolean less(Patron b) {
    Patron a = this;
    return a.patron_id < b.patron_id;
}

// return a string representation of the invoking object
public String toString() {
    return patron_id + " " + first_name + " " + last_name + " " + email " "
+ phone_number " " + class_name " " + instructor_name;;
}

// sample client
public static void main(String[] args) {

    // number of Patrons
    int n = Integer.parseInt(args[0]);

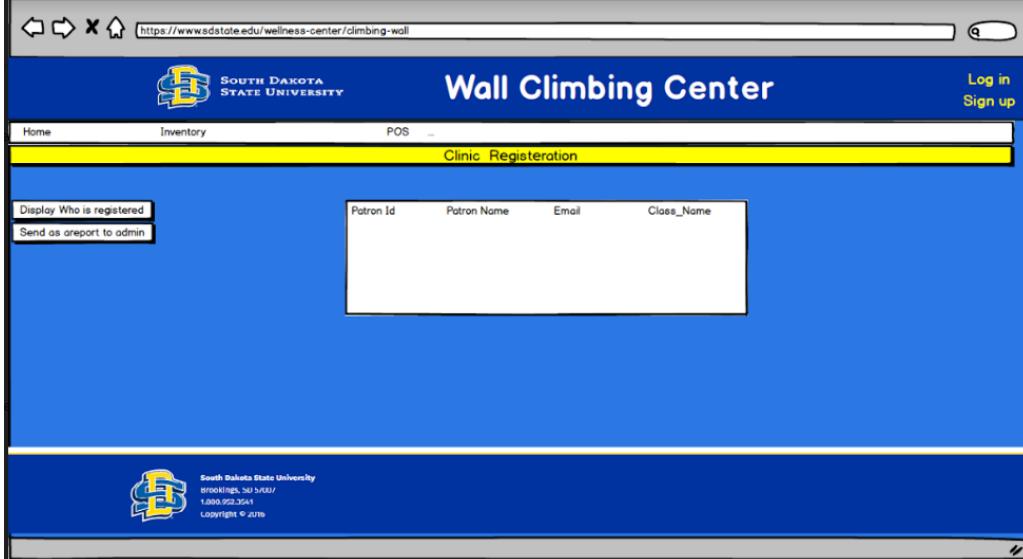
    // initialize an arary that holds n objects of type Patron
    Patron[] Patrons = new Patron[n];

    // read in the data
    for (int i = 0; i < n; i++) {
        String first_name = StdIn.readString();
        String last_name = StdIn.readString();
        String email = StdIn.readString();
        int patron_id = StdIn.readInt();
        int phone_number =StdIn.readInt();
        String class_name = StdIn.readString();

        String instructor_name = StdIn.readString();
        Patrons[i] = new Patron(first_name, last_name, email,
patron_id,phone_number,class_name,instructor_name);
    }

    // insertion sort Patrons in ascending order of patron_id
    for (int i = 0; i < n; i++) {
        for (int j = i; j > 0; j--) {
            if (Patrons[j].less(Patrons[j-1])) {
                Patron swap = Patrons[j];
                Patrons[j] = Patrons[j-1];
                Patrons[j-1] = swap;
            }
        }
    }
}

```

	<pre>         }     }      // print results     for (int i = 0; i &lt; n; i++) {         StdOut.println(Patrons[i]);     } }  } </pre>
Screen shot	 <p>The screenshot shows a web browser window for the "Wall Climbing Center" at <a href="https://www.sdsstate.edu/wellness-center/climbing-wall">https://www.sdsstate.edu/wellness-center/climbing-wall</a>. The page has a blue header with the SDSU logo and the text "WALL CLIMBING CENTER". It features a navigation bar with links for Home, Inventory, POS, Clinic, and Registration. Below the navigation, there are two buttons: "Display Who is registered" and "Send as a report to admin". A table is present with columns for Patron Id, Patron Name, Email, and Class_Name. At the bottom, there is a footer with the SDSU logo and contact information.</p>

#### 1.1.2.2.1.2 EMS system

WBS No. 1.1.2.2.1.2 EMS System	
Description	Display Scheduling System for appointment, Class Timings and the notification remainder and soon. The admin will have the full access to the EMS system.
Input	The input will be from the admin to show the available timing based on the date, day and the time and update the schedules to the database
Output	The available schedules will be the limited access available timings for the employee for the class timings , notification reminders and that will be displayed on the patron page.

Flowchart	<pre> graph TD     Start([Start]) --&gt; When[When EMS System Called]     When --&gt; Access{Access Specifier}     Access -- Employee --&gt; DisplayOld[/Display Old schedules/]     Access -- Admin --&gt; Admin[Admin]     Access -- Patron --&gt; DisplaySchedules[Display schedules]     DisplayOld --&gt; EditCreate{Edit or Create}     EditCreate -- Edit/update --&gt; EditNew[Edit to new Schedules]     EditNew --&gt; SendNotifications[Send Notifications]     SendNotifications --&gt; UpdatedDatabase([Updated database])     EditCreate -- Create --&gt; CreateNew[Create new Schedules sheet]     CreateNew --&gt; SendNotifications     SendNotifications --&gt; UpdatedDatabase   </pre>
pseudocode	<pre> import Date.*; import Time.*; public class Appointment {     private String location;     private String appointed;     private String purpose;     private int year;     private int month;     private int day;     private int hours;     private int minutes;     private int seconds;     public Appointment(int year, int month, int day, int hours, int minutes, String location, String appointed, String purpose) throws Exception     {         Date date = new Date(year, month, day);         Time time = new Time(hours, minutes);         this.year = year;         this.month = month;         this.day = day;         this.hours = hours;         this.minutes = minutes;         this.seconds = seconds;         this.location = location;         this.appointed = appointed;         this.purpose = purpose;     } }   </pre>

```

    }
    public Appointment() throws Exception
    {
        Date date = new Date();
        Time time = new Time();
        this.year = year;
        this.month = month;
        this.day = day;
        this.hours = hours;
        this.minutes = minutes;
        this.seconds = seconds;
        this.location = location;
        this.appointed = appointed;
        this.purpose = purpose;
    }
    public void setDate (int year, int month, int day) throws Exception
    {
        Date date = new Date(year, month, day);
        this.year = year;
        this.month = month;
        this.day = day;
        return;
    }
    public int getYear()
    {
        return year;
    }
    public int getMonth()
    {
        return month;
    }
    public int getDay()
    {
        return day;
    }
    public String getTime()
    {
        String result; //String representation of a Date
        result = "Hour=" + hours + "minutes=" + minutes;
        return result;
    }
    public void setTime (int hours, int seconds)throws Exception
    {
        Time time = new Time(hours, minutes);
        this.hours = hours;
        this.minutes = minutes;
        return;
    }
    public String getAppointed()
    {

```

```

        return appointed;
    }
    public String toString()
    {
        return "The appointment is on:" + day + "/" + month + "/" + year + "at " + hours + ":" +
        minutes + " in: " + location + " with " + appointed + " scheduled for " + purpose;
    }
    public int compareTo(Date otherDate)
    {
        if (getYear() != (otherDate.getYear()))
            return getYear() - (otherDate.getYear());
        else if (getMonth() != (otherDate.getMonth()))
            return getMonth() - (otherDate.getMonth());
        else
            return getDay() - (otherDate.getDay());
    }
    public static void main (String[]args) throws Exception
    {
        int comparisonResult;
        Appointment aAppt = new Appointment(2004, 2, 2, 11, 42, " , "Justin_Park", "Checkup");
        Appointment bAppt = new Appointment(2004, 2, 2, 12, 42, " , "Justin_Park", "Checkup");
        Appointment cAppt = new Appointment(2004, 4, 4, 1, 42, " , "Justin_Park", "Checkup");
        System.out.println("The appointment info : " + aAppt);
        System.out.println("The appointment info : " + bAppt);
        comparisonResult = aAppt.compareTo(bAppt);
        if (comparisonResult == 0)
            System.out.println(aAppt + "is equal to" + bAppt);
        else if (comparisonResult < 0)
            System.out.println(aAppt + "is less than" + bAppt);
        else
            System.out.println(aAppt + "is greater than" + bAppt);
    }
}

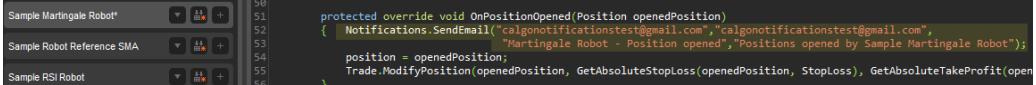
```

### 1.1.2.2.1.2.2 Send Notifications

#### WBS No. 1.1.2.2.1.2.2 Send notifications

Description	The send notifications will be done by the employee to the user page profile. Before that the user has to activate the notifications by valid verification of his email and the phone number of the user.
Input	The user will give the email and the phone number to send the 6 digit code for verification of the user from the communication system.

Output	The output will be the activate notification of the user successfully on the profile page.
Flow chart	<pre> graph TD     Start([Start]) --&gt; Decision1{Decision}     Decision1 --&gt; SendEmail[/Send an email/]     Decision1 --&gt; SendText[/Send a text/]     Decision1 --&gt; StaffAnnounce[/Announcement by staff/]     Decision1 --&gt; AdminAnnounce[/Announcement by admin/]     SendEmail --&gt; Converge1(( ))     SendText --&gt; Converge1     StaffAnnounce --&gt; Converge1     AdminAnnounce --&gt; Converge1     Converge1 --&gt; Decision2{Decision}     Decision2 --&gt; End([End])   </pre>
pseudocode	<pre> public class EmailSender{ public static void main(String args[]) { String to = "receive@abc.om"; // sender email String from = "sender@abc.com"; // receiver email String host = "127.0.0.1"; // mail server host Properties properties = System.getProperties(); properties.setProperty("mail.smtp.host", host) properties.setProperty("mail.smtp.host", host); Session session = Session.getDefaultInstance(properties); // default session try { MimeMessage message = new MimeMessage(session); // email message message.setFrom(new InternetAddress(from)); // setting header fields message.addRecipient(Message.RecipientType.TO, new InternetAddress(to)); message.setSubject("Test Mail from Java Program");   </pre>

	<pre> // subject line // actual mail body message.setText("You can send mail from Java program by using mail API, but you need" + "couple of more JAR files e.g. smtp.jar and activation.jar"); // Send message Transport.send(message); System.out.println("Email Sent successfully...."); } catch (MessagingException mex) { mex.printStackTrace(); } } } } </pre>
Screen shot	 <pre> 50 51     protected override void OnPositionOpened(Position openedPosition) 52     { 53         Notifications.SendEmail("calgonotificationtest@gmail.com", "Martingale Robot - Position opened", "Positions opened by Sample Martingale Robot"); 54         position = openedPosition; 55         Trade.ModifyPosition(openedPosition, GetAbsoluteStopLoss(openedPosition, StopLoss), GetAbsoluteTakeProfit(open 56     } </pre>

### 1.1.2.2.1.3 certification management

WBS No. 1.1.2.2.1.3 certification management	
Description	The certificate management will have the Track certification , validate certification and update validation status of the certification to the database by the employ
Input	Employ name, password, Patron name, password, patron ID and the Certificate of the patron in order to validate
Output	The certification status of the patron will be updated after the validation.
Pseudocode	<pre> public boolean isPDF(File file){     file = new File("Demo.pdf");     Scanner input = new Scanner(new FileReader(file));     while (input.hasNextLine()) {         final String checkline = input.nextLine();         if(checkline.contains("%PDF-")){             // a match!             return true;         }     }     return false; } </pre>

Screenshot	<p>The screenshot shows a web browser window for the "Wall Climbing Center". The URL is https://www.sdsstate.edu/wellness-center/climbing-wall. The page features a blue header with the South Dakota State University logo and the text "Wall Climbing Center". Below the header is a navigation bar with links for "Home", "Inventory", and "POS". A yellow horizontal bar spans across the middle of the page with the text "certification managements". On the left side of this bar are three buttons: "issue new certification", "update certification", and "check certificate validation". At the bottom of the page is a footer with the university logo and copyright information: "South Dakota State University", "ISSUED DATE: 5/23/2017", "1000.952.3541", and "Copyright © 2016".</p>
------------	--

#### **1.1.2.2.2.2 check in management**

WBS No. 1.1.2.2.2.2 Check In	
Description	The check- in system has very high priority for this system. Here the check In System will check whether the user/ patron has the access to the wall climbing center or not based on his Patron ID.
Validation	Patron's accessibility
Input	patron ID
Output	The output will be based on his Access to the rock climbing system. If it was accepted then he can enter the rock climbing center. If it was rejected then he will get the reason for reject.

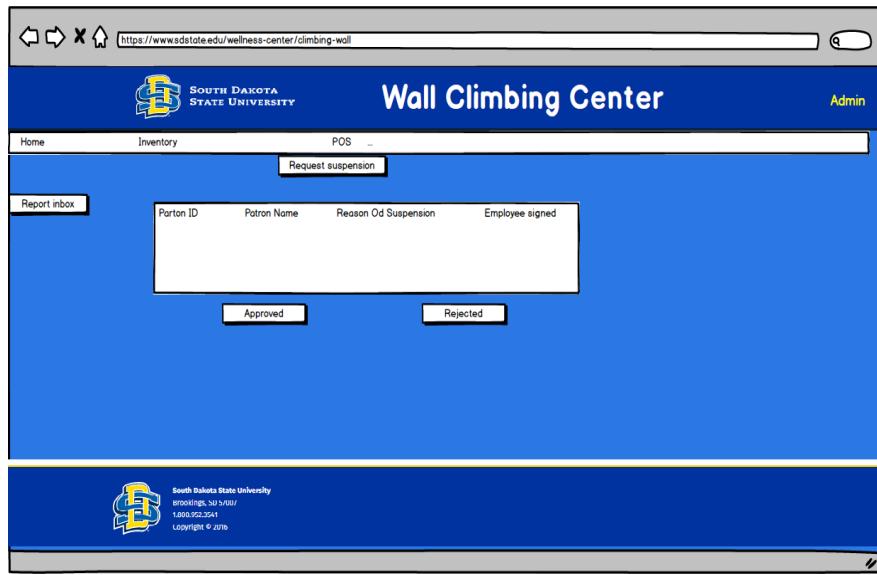
Flow chart	<pre> graph TD     Start([Start]) --&gt; When[When Check In]     When --&gt; Enter[Enter the Wall Climbing Center]     Enter --&gt; Registration{Validate patron Registration}     Registration -- Accepted --&gt; Duplication[Apply Duplication removal Algorithm based on Id No]     Duplication --&gt; Update[Update data based On date And month]     Update --&gt; Updated([Updated database])     Registration -- Rejected or Suspension --&gt; Inform[Inform Patron Cannot Check in Because of Suspension]     Registration -- Non registration reject --&gt; RegistrationSystem[Registration system]     RegistrationSystem --&gt; Validate{Validate}     Validate --&gt; Inform   </pre> <p>The flowchart illustrates the process of a patron checking in at a wall climbing center. It begins with a start node, followed by a decision point 'When Check In'. This leads to two parallel paths: one for entering the center and another for using equipment. The equipment path involves validating the patron's registration. If registration is accepted, it proceeds to remove duplicates and update the database. If rejected (due to suspension) or if non-registration, the patron is informed and the process ends. A separate 'Registration system' handles non-registration rejects.</p>
pseudocode	<pre> Patron_id[] addOne(Patron_id[] oldList) {     Patron_id[] newList = new Patron_id[oldList.length+1]; //if he/she new user will be added to newList      for(int i = 0; i &lt; oldList.length; i++)         newList[i] = oldList[i]; //if patron already exist get his/her details from oldList      return newList; // the last slot is available and == null }   </pre>

Screen shot	
-------------	--

### **1.1.2.2.2.5 request admin for user suspension**

<b>WBS No. 1.1.2.2.2.5 request admin for user suspension</b>	
Description	This function will do process the employee to request user for the patron suspension based on the not following the policies Suspension
Validation	Suspension
Input	Will be user id and user name and employee who request the suspension and reason of suspension
Output	The output will be approval and do process of suspension or reject the suspension request to employee

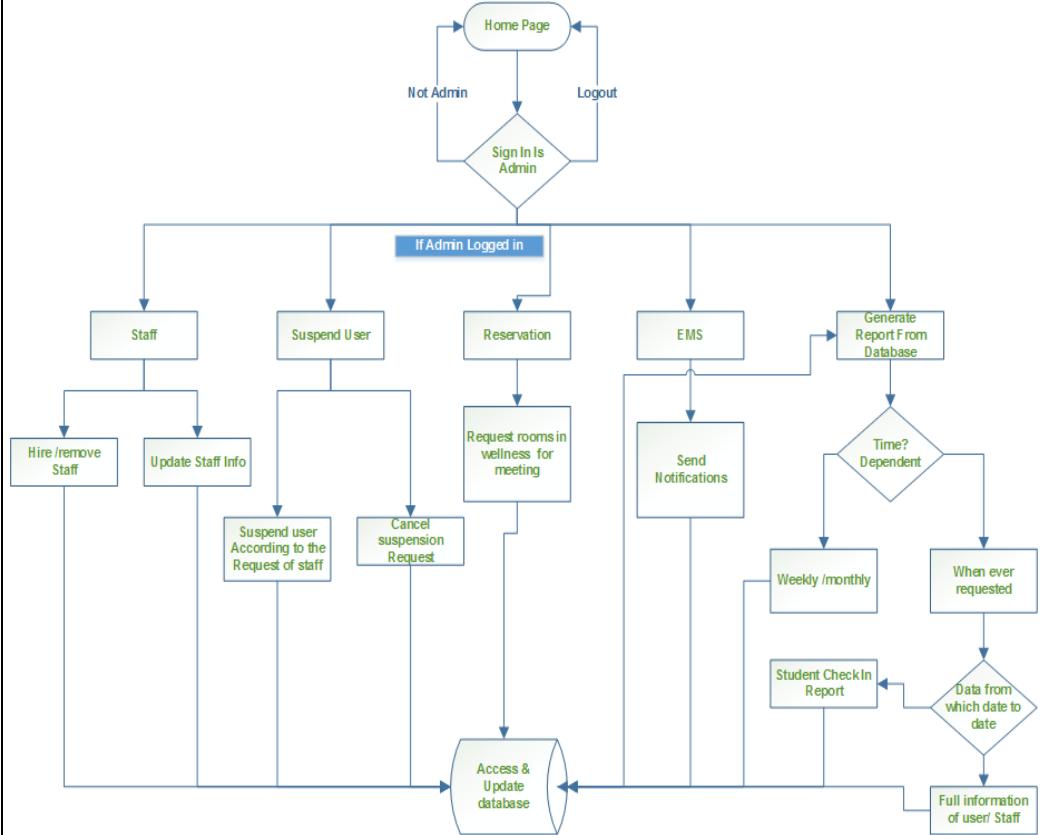
Screen shot



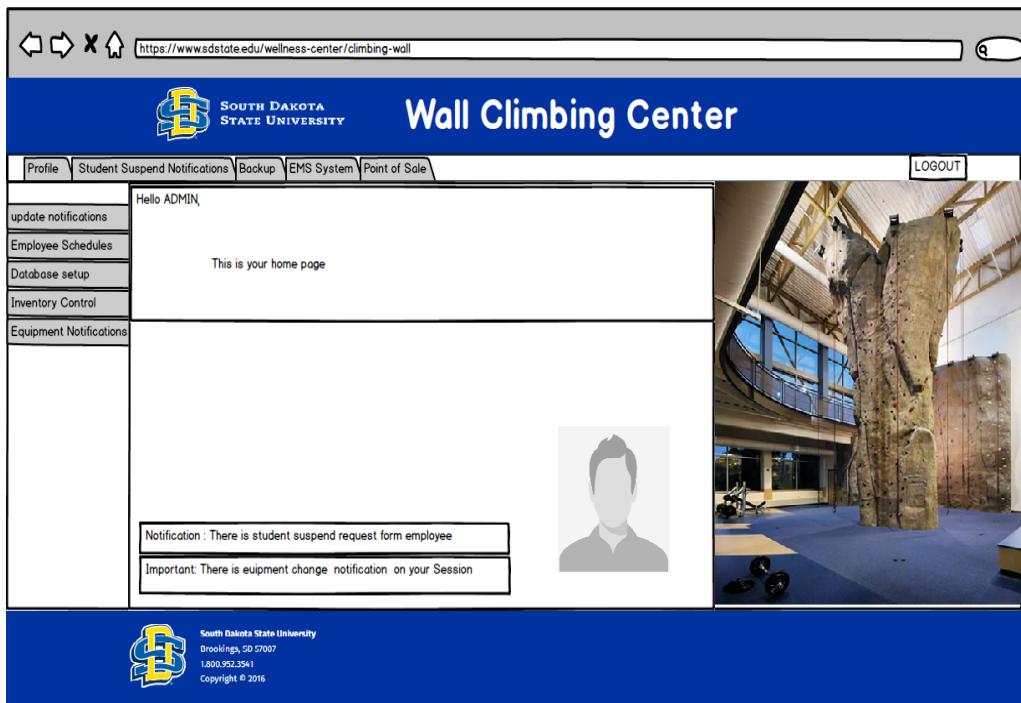
### 1.1.2.3 admin

WBS No. 1.1.2.3 admin	
Description	The admin Page has full access for the system. He can remove or add anyone to the system and generate weekly report or whatever he want to modify he can change it
Input	The input will be the admin login and the password and submit action.
Output	The admin profile page will be opened and the admin will get the full access to the system of rock climbing.

## Flow Chart



## Screen shot



### 1.1.2.3.1 staff management

WBS No. 1.1.2.3.1 Staff management	
Description	This function will let admin to add or delete an employee
Input	<p>Input for adding new employee First_name,Last_name,DOB,Sex,Address,Phone,Email Experience,Waiver form,Certification.</p> <p>For deleting an employee by entering employee name</p>
Output	A message of that indicate the addition or deletion of an employee
pseudocode	<pre> package ers.database;  import java.sql.ResultSet; import java.sql.SQLException;  public class Employees extends Database {      public void createNewEmployee(String firstName, String secondName, String emailAddress, String jobRole, Integer currentStatus, Integer hireType, Double wages) {         Integer employeeID;         try {             String getRowCountSQL = "SELECT COUNT(*) AS rowcount FROM employees";             ResultSet getCountResultSet = statement.executeQuery(getRowCountSQL);              getCountResultSet.next();             employeeID = getCountResultSet.getInt("rowcount") + 1;             getCountResultSet.close();              String createNewEmployeeSQL = "INSERT INTO employees (ID, first_name, second_name, email_address, job_role, current_status, hire_type, wage) VALUES (?, ?, ?, ?, ?, ?, ?, ?)";             preparedStatement = db.prepareStatement(createNewEmployeeSQL);             preparedStatement.setInt(1, employeeID);             preparedStatement.setString(2, firstName);             preparedStatement.setString(3, secondName);             preparedStatement.setString(4, emailAddress);             preparedStatement.setString(5, jobRole);             preparedStatement.setInt(6, currentStatus);             preparedStatement.setInt(7, hireType);             preparedStatement.setDouble(8, wages);              if (preparedStatement.executeUpdate() == 1) {                 System.out.println("Employee added successfully!");             } else {                 System.out.println("Employee addition failed.");             }         } catch (SQLException e) {             e.printStackTrace();         }     } } </pre>

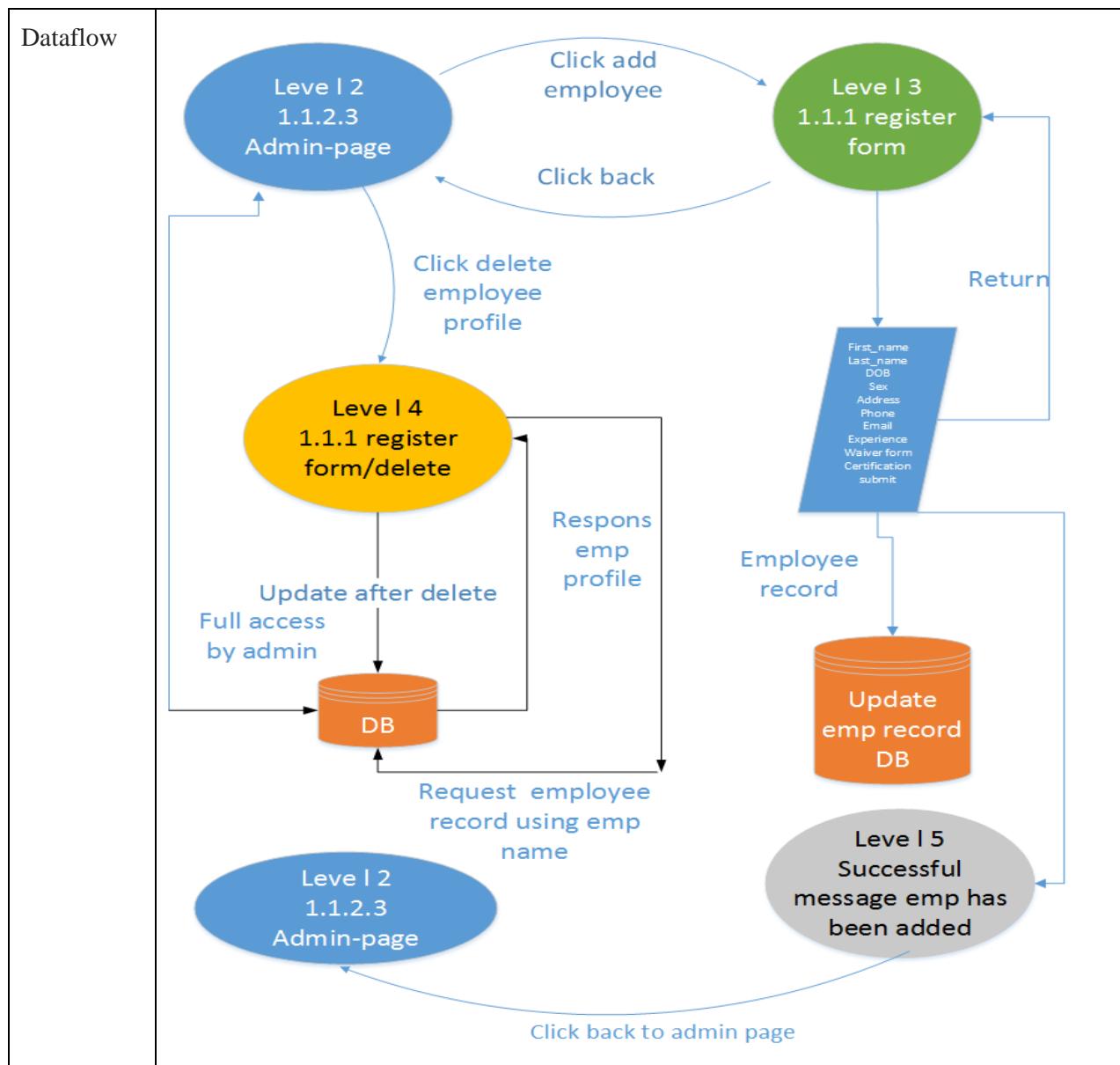
```

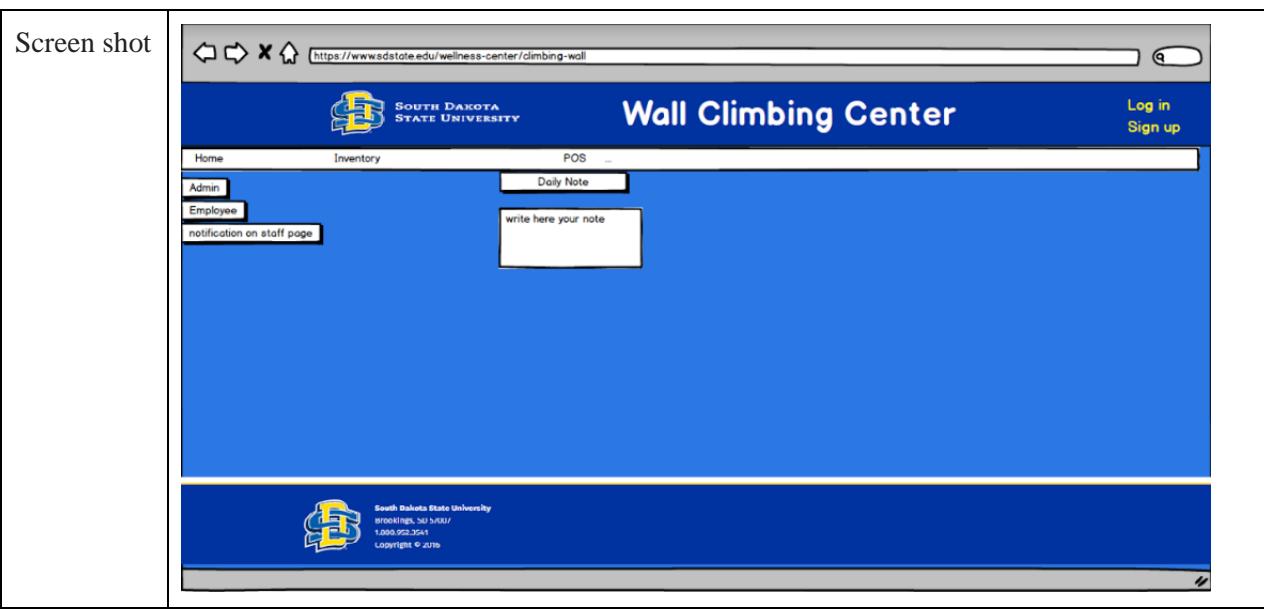
        System.out.println("Employee record added to system.");
    } else {
        System.err.println("Error adding new employee record to system.");
    }
} catch (Exception ex) {
    System.err.println(ex.toString());
} finally {
    tryClosePreparedStatement();
    tryCloseConnectionToDatabase();
}
}

public void deleteEmployee(Integer employeeID) {
try {
    String deleteEmployeeSQL = "DELETE FROM employees where ID = ?";
    preparedStatement = db.prepareStatement(deleteEmployeeSQL);
    preparedStatement.setInt(1, employeeID);

    if (preparedStatement.executeUpdate() == 1) {
        System.out.println("Employee record deleted.");
    } else {
        System.err.println("Error deleted employee record.");
    }
} catch (SQLException ex) {
    System.err.println(ex.toString());
} finally {
    tryClosePreparedStatement();
    tryCloseConnectionToDatabase();
}
}
}

```





### 1.1.2.3.2 admin for user suspension

WBS No. 1.1.2.3.2 admin for user suspension	
Description	This function will do process of user suspension
Input	Will be user id and user name and employee who request the suspension and reason of suspension
Output	The output will be approval and do process of suspension or reject the suspension request to employee
Pseudocode	<pre>If (LOGIN==ADMIN){      Check Notification= j. Button (Show text notification);     If(employ request_patron ID==1)     {         Db.delete_access_patron ID = Curr_time+ 7days;     }     If(employ request_patron ID==2)     {         Db.delete_access_patron ID = Curr_time+ 30days;     }     If(employ request_patron ID==3)     {         Db.delete_access_patron ID = Curr_time+ 90days;     }     If(employ request_patron ID==4)</pre>

	<pre>{ Db.delete_access_patron ID = Curr_time+ 365days; } }</pre>
--	---

### 1.1.2.3.3 Reservation

WBS No. 1.1.2.3.3 Reservation	
Description	The reservation system will be accessed by the administrator to book the system with the place reason, time and the date for organizing meeting in the wellness center
Input	The input will be from the admin about the Timings available for the meetings
Output	The reservation will be done at the timings available in the SDSU wellness center.
Pseudocode	<pre>public class Appointment { private String location; private String appointed; private String purpose; private int year; private int month; private int day; private int hours; private int minutes; private int seconds; public Appointment(int year, int month, int day, int hours, int minutes, String location, String appointed, String purpose) throws Exception { Date date = new Date(year, month, day); Time time = new Time(hours, minutes); this.year = year; this.month = month; this.day = day; this.hours = hours; this.minutes = minutes; this.seconds = seconds; this.location = location; this.appointed = appointed; this.purpose = purpose; }</pre>

```

public Appointment() throws Exception
{
    Date date = new Date();
    Time time = new Time();
    this.year = year;
    this.month = month;
    this.day = day;
    this.hours = hours;
    this.minutes = minutes;
    this.seconds = seconds;
    this.location = location;
    this.appointed = appointed;
    this.purpose = purpose;
}
public void setDate (int year, int month, int day) throws Exception
{
    Date date = new Date(year, month, day);
    this.year = year;
    this.month = month;
    this.day = day;
    return;
}
public int getYear()
{
    return year;
}
public int getMonth()
{
    return month;
}
public int getDay()
{
    return day;
}
public String getTime()
{
    String result; //String representation of a Date
    result = "Hour=" + hours + "minutes=" + minutes;
    return result;
}
public void setTime (int hours, int seconds)throws Exception
{
    Time time = new Time(hours, minutes);
    this.hours = hours;
    this.minutes = minutes;
    return;
}
public String getAppointed()
{
    return appointed;
}

```

```

}
public String toString()
{
return "The appointment is on:" + day + "/" + month + "/" + year + "at " + hours + ":" +
minutes + " in: " + location + " with " + appointed + " scheduled for " + purpose;
}
public int compareTo(Date otherDate)
{
if (getYear() != (otherDate.getYear()))
return getYear() - (otherDate.getYear());
else if (getMonth() != (otherDate.getMonth()))
return getMonth() - (otherDate.getMonth());
else
return getDay() - (otherDate.getDay());
}
public static void main (String[]args) throws Exception
{
int comparisonResult;
Appointment aAppt = new Appointment(2004, 2, 2, 11, 42, " , "Justin_Park", "Checkup");
Appointment bAppt = new Appointment(2004, 2, 2, 12, 42, " , "Justin_Park", "Checkup");
Appointment cAppt = new Appointment(2004, 4, 4, 1, 42, " , "Justin_Park", "Checkup");
System.out.println("The appointment info : " + aAppt);
System.out.println("The appointment info : " + bAppt);
comparisonResult = aAppt.compareTo(bAppt);
if (comparisonResult == 0)
System.out.println(aAppt + "is equal to" + bAppt);
else if (comparisonResult < 0)
System.out.println(aAppt + "is less than" + bAppt);
else
System.out.println(aAppt + "is greater than" + bAppt);
}
}

```

#### 1.1.2.3.4 EMS system

WBS No. 1.1.2.3.4 EMS System	
Description	Display Scheduling System for appointment, Class Timings and the notification remainder and soon. The admin will have the full access to the EMS system.
Input	The input will be from the admin to show the available timing based on the date, day and the time and update the schedules to the database
Output	The available schedules will be the limited access available timings for the employee for the class timings , notification reminders and that will be displayed on the patron page.

Flow chart	<pre> graph TD     Start([Start]) --&gt; When[When EMS System Called]     When --&gt; Access{Access Specifier}     Access -- Employee --&gt; DisplayOld[/Display Old schedules/]     Access -- Patron --&gt; DisplaySchedules[Display schedules]     Access -- "Public Access Any time" --&gt; Admin[Admin]     DisplayOld --&gt; EditCreate{Edit or Create}     EditCreate -- Create --&gt; CreateSheet[Create new Schedules sheet]     CreateSheet --&gt; Send[Send Notifications]     EditCreate -- "Edit/update" --&gt; EditSchedules[Edit to new Schedules]     EditSchedules --&gt; Send     Send --&gt; UpdatedDatabase([Updated database])     Admin --&gt; UpdatedDatabase   </pre>
Pseudocode	<pre> public class Appointment {     private String location;     private String appointed;     private String purpose;     private int year;     private int month;     private int day;     private int hours;     private int minutes;     private int seconds;     public Appointment(int year, int month, int day, int hours, int minutes, String location, String appointed, String purpose) throws Exception     {         Date date = new Date(year, month, day);         Time time = new Time(hours, minutes);         this.year = year;         this.month = month;         this.day = day;         this.hours = hours;         this.minutes = minutes;         this.seconds = seconds;         this.location = location;         this.appointed = appointed;         this.purpose = purpose;     }     public Appointment() throws Exception   </pre>

```

{
Date date = new Date();
Time time = new Time();
this.year = year;
this.month = month;
this.day = day;
this.hours = hours;
this.minutes = minutes;
this.seconds = seconds;
this.location = location;
this.appointed = appointed;
this.purpose = purpose;
}
public void setDate (int year, int month, int day) throws Exception
{
Date date = new Date(year, month, day);
this.year = year;
this.month = month;
this.day = day;
return;
}
public int getYear()
{
return year;
}
public int getMonth()
{
return month;
}
public int getDay()
{
return day;
}
public String getTime()
{
String result; //String representation of a Date
result = "Hour=" + hours + "minutes=" + minutes;
return result;
}
public void setTime (int hours, int seconds)throws Exception
{
Time time = new Time(hours, minutes);
this.hours = hours;
this.minutes = minutes;
return;
}
public String getAppointed()
{
return appointed;
}

```

```

public String toString()
{
    return "The appointment is on:" + day + "/" + month + "/" + year + "at " + hours + ":" +
minutes + " in: " + location + " with " + appointed + " scheduled for " + purpose;
}
public int compareTo(Date otherDate)
{
    if (getYear() != (otherDate.getYear()))
        return getYear() - (otherDate.getYear());
    else if (getMonth() != (otherDate.getMonth()))
        return getMonth() - (otherDate.getMonth());
    else
        return getDay() - (otherDate.getDay());
}
public static void main (String[]args) throws Exception
{
    int comparisonResult;
    Appointment aAppt = new Appointment(2004, 2, 2, 11, 42, " ", "Justin_Park", "Checkup");
    Appointment bAppt = new Appointment(2004, 2, 2, 12, 42, " ", "Justin_Park", "Checkup");
    Appointment cAppt = new Appointment(2004, 4, 4, 1, 42, " ", "Justin_Park", "Checkup");
    System.out.println("The appointment info : " + aAppt);
    System.out.println("The appointment info : " + bAppt);
    comparisonResult = aAppt.compareTo(bAppt);
    if (comparisonResult == 0)
        System.out.println(aAppt + "is equal to" + bAppt);
    else if (comparisonResult < 0)
        System.out.println(aAppt + "is less than" + bAppt);
    else
        System.out.println(aAppt + "is greater than" + bAppt);
}
}

```

Screenshot	<p><b>Details</b></p> <p>Add a title for the event</p> <p>Add a location <input type="button" value="Add room"/></p> <p>Start  <input type="text" value="Mon 10/24/2016"/> <input type="text" value="9:00 AM"/> <input type="checkbox"/> All day</p> <p>End  <input type="text" value="Mon 10/24/2016"/> <input type="text" value="9:30 AM"/> <input type="checkbox"/> Private</p> <p>Repeat  <input type="text" value="Never"/> Save to calendar <input type="text" value="Calendar"/></p> <p>Reminder  <input type="text" value="15 minutes"/> Show as <input type="text" value="Busy"/></p> <p><a href="#">Add an email reminder</a></p>
------------	---

#### 1.1.2.3.5 List Serv

WBS No. 1.1.2.3.6 List Serv	
Description	This function will let admin to get contact with patrons
Input	All eamil address of all patrons and employees
Output	Generate a list with all emails address for patrons and employees
Pseudocode	<pre>public class WriteExcelDemo {     public static void main(String[] args)     {         //Blank workbook         XSSFWorkbook workbook = new XSSFWorkbook();          //Create a blank sheet</pre>

```

XSSFSheet sheet = workbook.createSheet("listServ record");

//This data needs to be written (Object[])
Map<String, Object[]> data = new TreeMap<String, Object[]>();
data.put("1", new Object[] {"Patron_id", "Patron_name", "Email"});
data.put("2", new Object[] {7350, "Raju", "Rajo@jacks.com"});
data.put("2", new Object[] {7350, "Shaho", "shaho@jacks.com});

//Iterate over data and write to sheet
Set<String> keyset = data.keySet();
int rownum = 0;
for (String key : keyset)
{
    Row row = sheet.createRow(rownum++);
    Object [] objArr = data.get(key);
    int cellnum = 0;
    for (Object obj : objArr)
    {
        Cell cell = row.createCell(cellnum++);
        if(obj instanceof String)
            cell.setCellValue((String)obj);
        else if(obj instanceof Integer)
            cell.setCellValue((Integer)obj);
    }
}
try
{
    //Write the workbook in file system
    FileOutputStream out = new FileOutputStream(new
File("howtodoinjava_demo.xlsx"));
    workbook.write(out);
    out.close();
    System.out.println("howtodoinjava_demo.xlsx written successfully on disk.");
}
catch (Exception e)
{
    e.printStackTrace();
}
}
}

```

Patron report with total participation									
S-NO	First_name	Last_name	Email	phone	expiration date	attended	from_date	End_date	
1	Hussain	Otudi	Ot@jacks	605-592	12/16/16	40	10/10/2016	10/16/2016	
2	shaoh	zhang	Sh@jacks	605-444	12/17/16	35	10/10/2016	10/16/2016	
3	Lee	haney	Lee@jacks	605-232	12/18/16	40	10/10/2016	10/16/2016	
4	Shin	sung	Shin@jacks	605-232	12/19/16	32	10/10/2016	10/16/2016	
5	Rajo	appala	Rajo@jacks	605-232	12/20/16	27	10/10/2016	10/16/2016	
6	Sayun	sahu	Sayun@jacks	605-232	12/21/16	40	10/10/2016	10/16/2016	
7	Bader	alzeidi	Bader@jacks	605-232	12/22/16	25	10/10/2016	10/16/2016	
8	Ali	selhenia	Ali@jacks	605-232	12/23/16	20	10/10/2016	10/16/2016	
9	Min	manki	Min@jacks	605-232	12/24/16	15	10/10/2016	10/16/2016	
10	Hamer	george	Hamer@jacks	605-232	12/25/16	40	10/10/2016	10/16/2016	
13									
14									
15									

#### 1.1.2.3.6 report generation

WBS No. 1.1.2.3.6 Report Generation	
Description	The report management will provide the report of the wall climbers based on the admin requirement.
Input	The input will be the type of the report like unique participation, Demographic wall users , total participation, new users and time period, and the day date and patron ID after giving the type of the report needed.
Output	The return report will be stored on the system and will be displayed on the screen of the admin.
Pseudocode	<pre> public class WriteExcelDemo {     public static void main(String[] args)     {         //Blank workbook         XSSFWorkbook workbook = new XSSFWorkbook();          //Create a blank sheet         XSSFSheet sheet = workbook.createSheet("Employee Data");          //This data needs to be written (Object[])         Map&lt;String, Object[]&gt; data = new TreeMap&lt;String, Object[]&gt;();         data.put("1", new Object[] { "S-NO", "First_NAME",         "LAST_NAME", "Email", "ATTENDANC", "START_DATE", "END_DATE" });         data.put("2", new Object[] { 1, "SHAHO",         "Shukla", "SHAHO@JACKS", 33, 10/10/2016, 10/12/2016 });         data.put("3", new Object[] { 2, "RAJO",         "Rajo", "Rajo@jacks", 605-232, 12/20/16, 27 });     } }</pre>

```

"APALA","RAJO@JACKS",32,10/10/2016,10/12/2016});
    data.put("4", new Object[] {3, "HUSSAIN",
"OTUDI","HUSSAIN@JACKS",22,10/10/2016,10/12/2016});
    data.put("5", new Object[] {4, "BADER",
"ALZIDI","BADER@JACKS",40,10/10/2016,10/12/2016});

//Iterate over data and write to sheet
Set<String> keyset = data.keySet();
int rounum = 0;
for (String key : keyset)
{
    Row row = sheet.createRow(rounum++);
    Object [] objArr = data.get(key);
    int cellnum = 0;
    for (Object obj : objArr)
    {
        Cell cell = row.createCell(cellnum++);
        if(obj instanceof String)
            cell.setCellValue((String)obj);
        else if(obj instanceof Integer)
            cell.setCellValue((Integer)obj);
    }
}
try
{
    //Write the workbook in file system
    FileOutputStream out = new FileOutputStream(new
File("howtodoinjava_demo.xlsx"));
    workbook.write(out);
    out.close();
    System.out.println("howtodoinjava_demo.xlsx written successfully on disk.");
}
catch (Exception e)
{
    e.printStackTrace();
}
}
}

```

Patron report with unique participation									
S-NO	First_name	Last_name	Email	phone	expiration date	attended	from_date	End_date	
1	Hussain	Otudi	Ot@jacks	605-592	12/16/16	40	10/10/2016	10/16/2016	
2	shaoh	zhang	Sh@jacks	605-444	12/17/16	35	10/10/2016	10/16/2016	
3	Lee	haney	Lee@jacks	605-232	12/18/16	40	10/10/2016	10/16/2016	
4	Shin	sung	Shin@jacks	605-232	12/19/16	32	10/10/2016	10/16/2016	
5	Rajo	appala	Rajo@jacks	605-232	12/20/16	27	10/10/2016	10/16/2016	
6	Sayun	sahu	Sayun@jacks	605-232	12/21/16	40	10/10/2016	10/16/2016	
7	Bader	alzeidi	Bader@jacks	605-232	12/22/16	25	10/10/2016	10/16/2016	
8	Ali	selhenia	Ali@jacks	605-232	12/23/16	20	10/10/2016	10/16/2016	
9	Min	manki	Min@jacks	605-232	12/24/16	15	10/10/2016	10/16/2016	
10	Hamer	george	Hamer@jacks	605-232	12/25/16	40	10/10/2016	10/16/2016	
11									
12									
13									
14									

## 1.2 inventory

WBS No. 1.2 inventory	
Description	This function will allow admin to track item based on id and purchase date, retirement date and replacement cost
Input	The input will be item_id and item_name
Output	The output will be a notification that inform admin about item when is getting close to retirement
Flow chart	<pre> graph TD     Home([Home]) --&gt; InvControl[Inventory control]     InvControl --&gt; ClimbingLevel[Level of climbing]     InvControl --&gt; WallInfo[Information Of wall climbing]     InvControl --&gt; InternetSearch[Internet Search]     InternetSearch --&gt; SearchEngine[Search Engine]     ClimbingLevel --&gt; Eligibility{Eligibility Criteria}     WallInfo --&gt; EquipmentInfo[Equipment Information]     EquipmentInfo --&gt; Database((Database))     Database --&gt; EquipmentInfo     SearchEngine --&gt; Database     Eligibility -- Not eligible --&gt; Certification{Certification}     Certification --&gt; ShouldHelp[Should need help]     Eligibility -- Eligible --&gt; CanClimb[Can climb individual]     </pre> <p>The flowchart illustrates the process for managing climbing equipment. It starts at the Home screen, which leads to the Inventory control module. From there, it branches into three main paths: Level of climbing, Information Of wall climbing, and Internet Search. The Internet Search path leads to a Search Engine. The Level of climbing path leads to an Eligibility Criteria decision diamond. If the criteria are not met, it leads to a Certification decision diamond, which then leads to a 'Should need help' outcome. If the criteria are met, it leads to a 'Can climb individual' outcome. The Information Of wall climbing path leads to Equipment Information, which then connects to a Database.</p>
	<pre> public class WriteExcelDemo {     public static void main(String[] args)     {         // Implementation details     } } </pre>

```

{
    //Blank workbook
    XSSFWorkbook workbook = new XSSFWorkbook();

    XSSFSheet sheet = workbook.createSheet("Inventory Data");

    //This data needs to be written (Object[])
    Map<String, Object[]> data = new TreeMap<String, Object[]>();
    data.put("1", new Object[] {"Item_id",
        "Item_name", "Cost", "Item_condition", "Purshase_date", "Expriation_date"});
    data.put("2", new Object[] {1, "Rope_Harnes",
        $155, "New",33,10/10/2016,10/12/2017});

    //Iterate over data and write to sheet
    Set<String> keyset = data.keySet();
    int rounum = 0;
    for (String key : keyset)
    {
        Row row = sheet.createRow(rounum++);
        Object [] objArr = data.get(key);
        int cellnum = 0;
        for (Object obj : objArr)
        {
            Cell cell = row.createCell(cellnum++);
            if(obj instanceof String)
                cell.setCellValue((String)obj);
            else if(obj instanceof Integer)
                cell.setCellValue((Integer)obj);
        }
    }
    try
    {
        //Write the workbook in file system
        FileOutputStream out = new FileOutputStream(new
File("howtodoinjava_demo.xlsx"));
        workbook.write(out);
        out.close();
        System.out.println("howtodoinjava_demo.xlsx written successfully on disk.");
    }
    catch (Exception e)
    {
        e.printStackTrace();
    }
}

```

Screen shot

No	Item	Start date	End date	Price
1	Rope Climbing Harness	10/9/2016	11/9/2016	\$99
2	Climbing Harness	10/9/2016	11/9/2016	\$299
3	Chest Harness	10/9/2016	11/9/2016	\$145
4	Climbing Shoes	10/9/2016	11/9/2016	\$199

## 7. Appendix

### Appendix A. Data dictionary

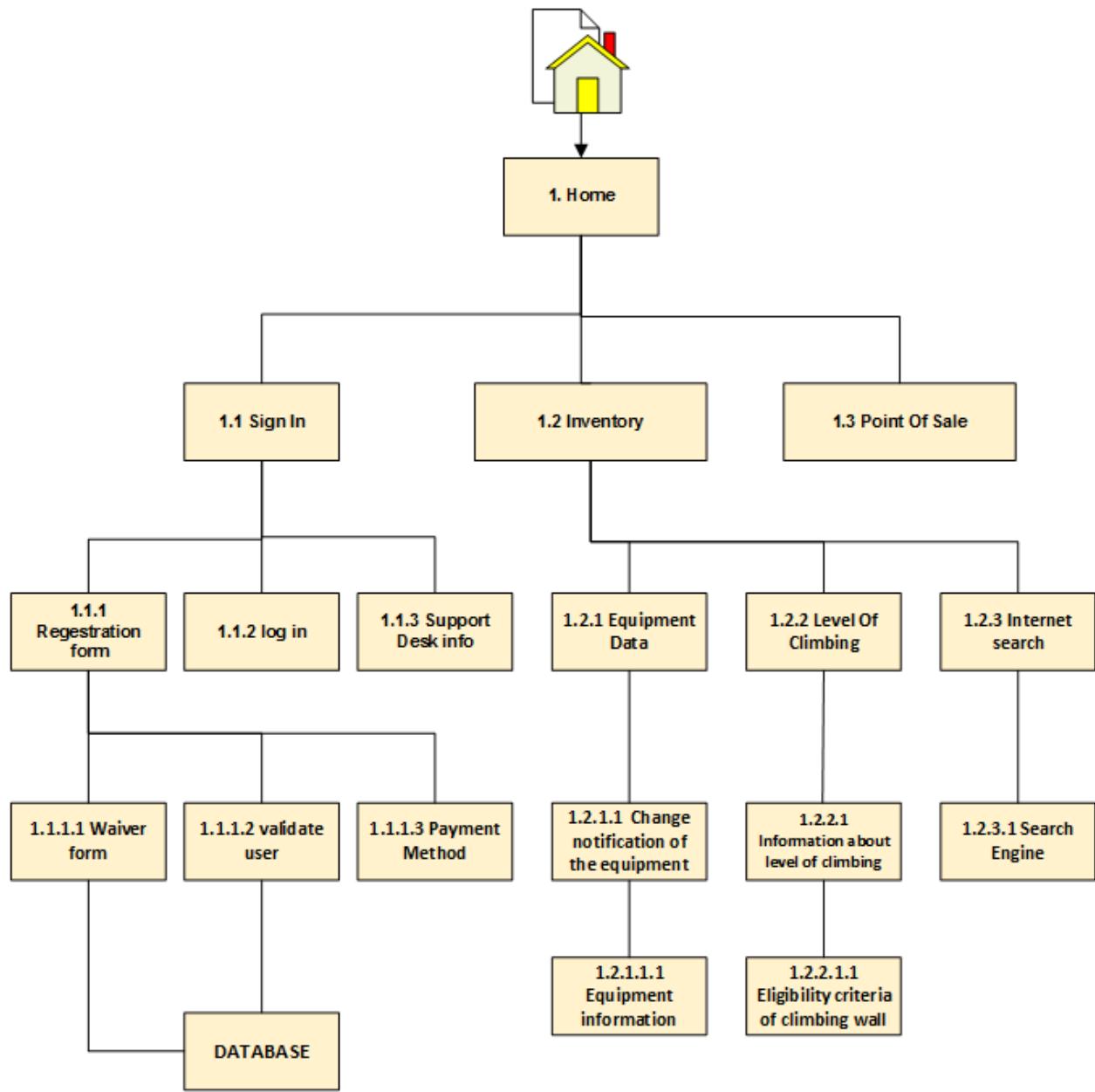
Appendix A. Data Dictionary					
Variable Name	Variable Type	Length	WBS NO	Valid range	Comments
First_name	String	2-25	WBS1.1.1 WBS1.1.2.3.1	a.....z A.....Z	the first name of user
last_name	String	2-25	WBS 1.1.1 WBS1.1.2.3.1	a.....z A.....Z	the last name of user
email	String	50	WBS1.1.1 WBS1.1.1.4 WBS1.1.2.2.1.1 WBS 1.1.2.3.1  WBS 1.1.2.2.1.2.2  WBS 1.1.2.3.5 WBS 1.1.1.2	a.....z A.....Z - .0.....9	email address
confirm_email	String	50	WBS1.1.1	a.....z A.....Z -	email address

				@ .0.....9	
password	String	6-12	WBS 1.1.1 WBS1.1.2 WBS1.1.2.2 WBS 1.1.2.2.1.3 WBS1.1.2.3	0.....9 a.....z A.....Z (/,*,- ,+.,!,@,#,\$,%,&,&?)	password
Confirm password	String	6-12	WBS1.1.1	0....9 a.....z A.....Z (/,*,- ,+.,!,@,#,\$,%,&,&?)	password
phone	String	10	WBS1.1.1 WBS1.1.1.4 WBS1.1.2.2.1.1 WBS1.1.2.2.1.2.2 WBS1.1.2.3.1	0.....9	phone number
DOB	String	20	WBS1.1.1 WBS1.1.2.3.1	0.....9 /	the date of birth
sex	String	10	WBS1.1.1 WBS1.1.2.3.1	male female	Male or female
Patron name	String	2-25	WBS1.1.1.1 WBS1.1.2.2.1.3	a.....z A.....Z	patron name
PatronID	String	6	WBS1.1.1.1 WBS1.1.2.2.1.3 WBS1.1.2.2.2.2 WBS1.1.2.2.1.1	0.....9	the patron ID
address	String	100	WBS1.1.1.4	a.....z A.....Z 0.....9	user address
user_ID	String	6	WBS1.1.2 WBS1.1.2.2.2.5 WBS1.1.2.3.2 WBS1.1.2.3.6	0.....9	user ID
Employee_name	String	2-25	WBS1.1.2.2 WBS1.1.2.2.1.3	a.....z A.....Z	employee name

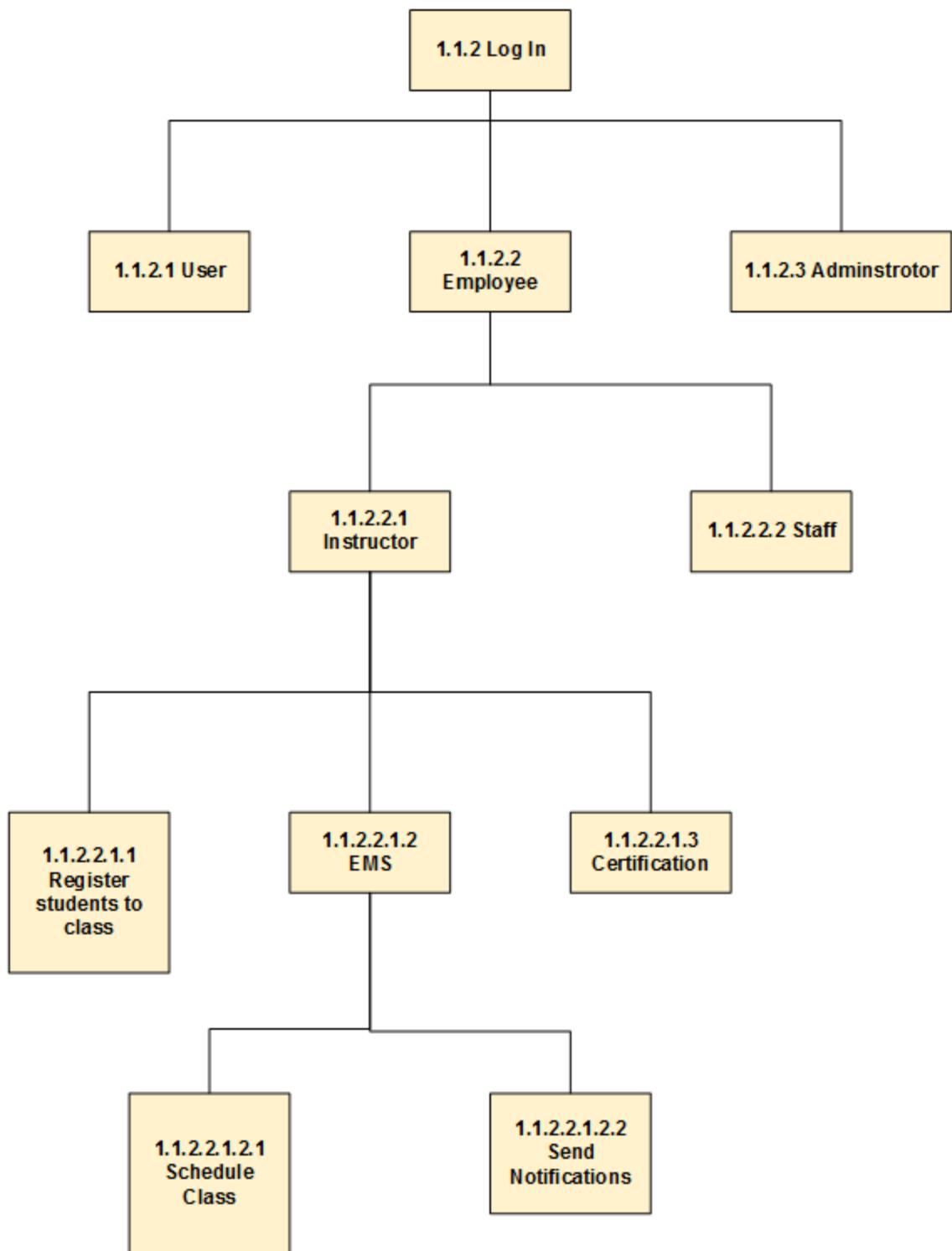
			WBS1.1.2.2.2.5 WBS1.1.2.3.2		
student_name	String	2-25	WBS1.1.2.2.1.1	a.....z A.....Z	student name
Instructor_name	String	2-25	WBS1.1.2.2.1.1 WBS1.1.2.2.1.2.1	a.....z A.....Z	instructor's name
time	String	20	WBS1.1.2.2.1.2.1 WBS1.1.2.3.3 WBS1.1.2.3.4 WBS1.1.2.3.6	am,pm,::0.9	the time
date	String	20	WBS1.1.2.2.1.2.1 WBS1.1.2.3.4 WBS1.1.2.3.6	a.....z A.....Z 0.....9 /,	the date
room	String	20	WBS1.1.2.2.1.2.1	a.....z A.....Z 0.....9	room number
User name	String	2-25	WBS1.1.2.2.2.5 WBS1.1.2.3.2 WBS1.1.2.3.6 WBS1.1.2.3.6	a.....z A.....Z	Enter the user name
Reason of suspension	String	500	WBS1.1.2.2.2.5 WBS1.1.2.3.2	text	Enter the reason
Admin name	String	2-25	WBS1.1.2.3	a.....z A.....Z	Enter the admin name
employee_ID	String	20	WBS1.1.2.3.6	0.....9	Enter the employee ID
item name	String	2-25	WBS1.2	a.....z A.....Z	Enter the item name
item_ID	String	20	WBS1.2	0.....9	Enter the item ID

## Appendix B. FWBS

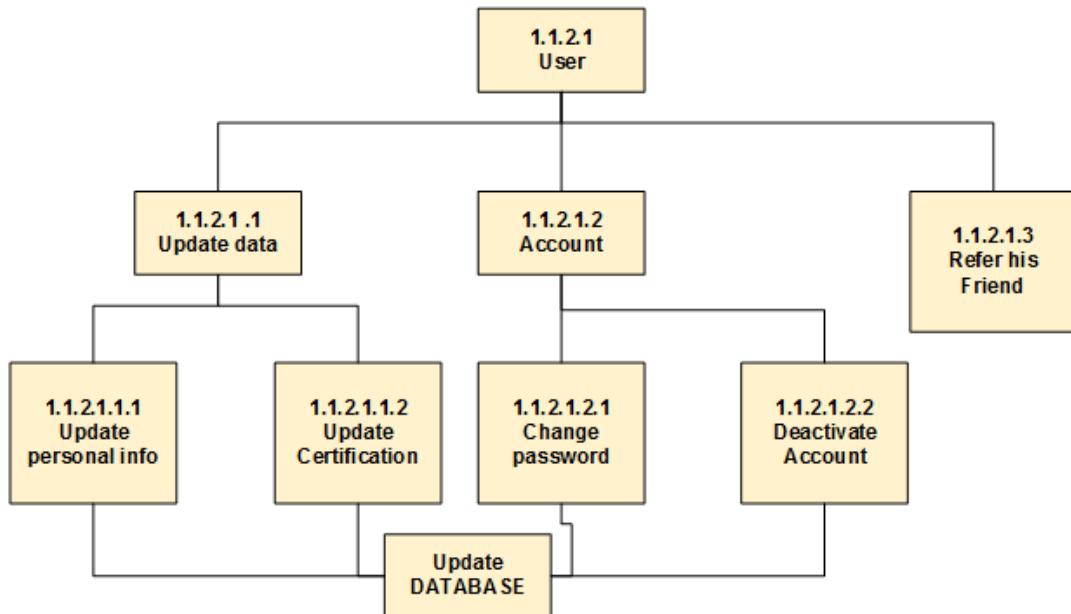
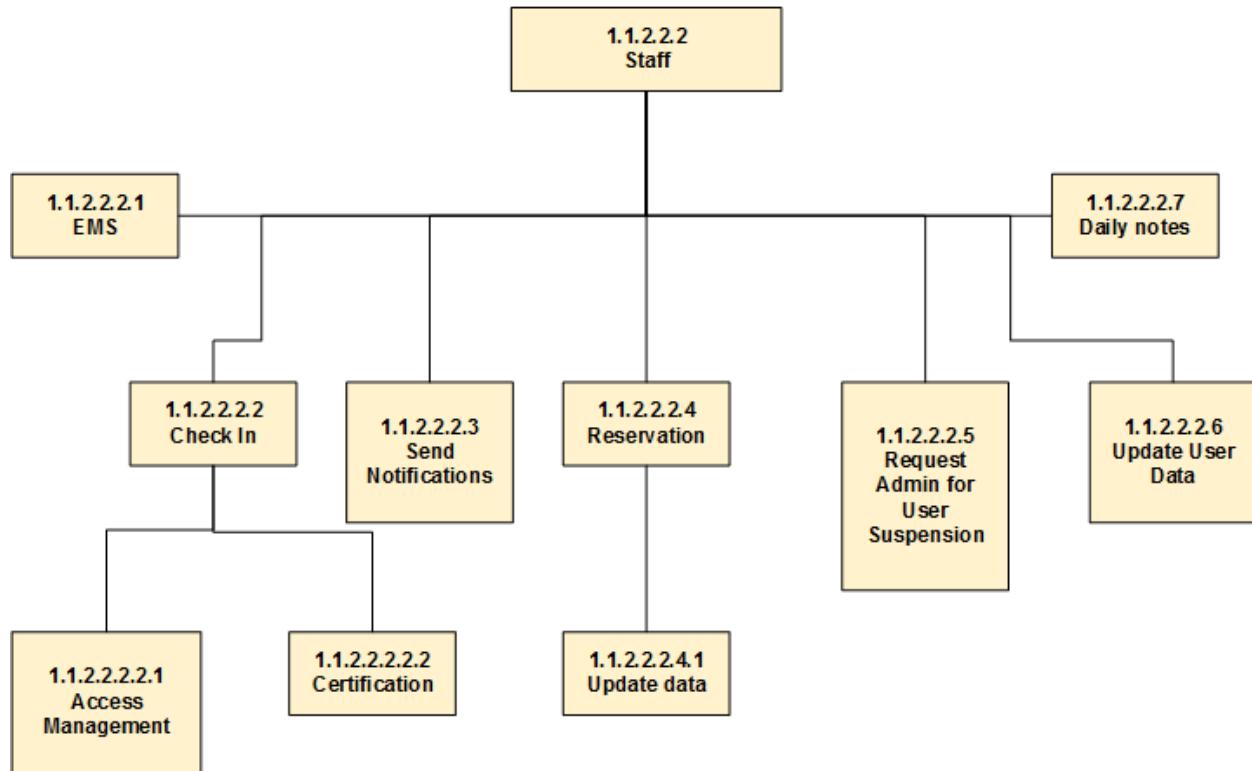
- Main Page:



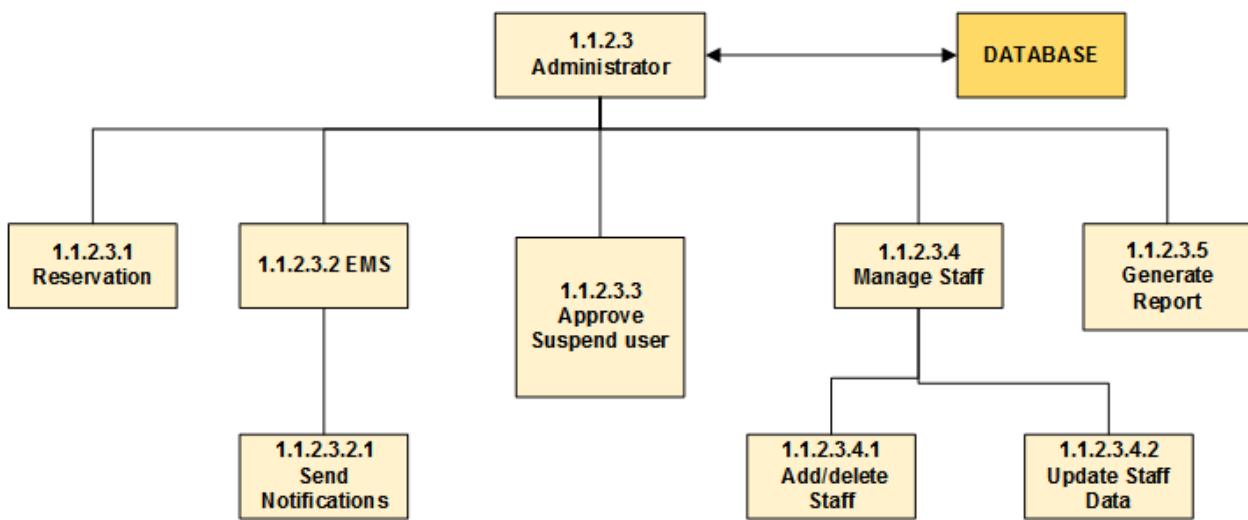
- Login Page:



- Staff and User



- Admin:



## Appendix C. Meeting Log

**10/10 2pm-4pm Mon DD plan**                           **WENS lab**                           **2 hours**

Attendee: Shaohu, Appala, Hussein , Bader

Task:

Drop EMS

Github account

Netbeans--desktop application

textbook:Netbeans IDE 8 cookbook

DD plan

**10/12 9-9:15 PM Wed Daily Scrum meeting**                           **WENS lab**                           **15 mins**

Attendee: Shaohu, Appala, Hussein

Task:

DFD discussion

**10/13 9-9:15 PM Thur Daily Scrum meeting**                           **WENS lab**                           **15 mins**

Attendee: Shaohu, Appala, Hussein, Bader

Tasks:

- DFD discussion
- Split work

**10/14 9-9:15 AM Fri Daily Scrum meeting**                           **WENS lab**                           **15 mins**

Attendee: Shaohu, Appala, Hussein

Task:

- Finish all DFD by midnight
- Design data dictionary table

**10/16 9-9:15 AM Sun Daily Scrum meeting**

**Attendee: Shaohu, Appala, Hussein**

**Task:**

- Review DFD

**WENS lab 15 mins**

**10/17 9-9:15 AM Mon Daily Scrum meeting**

**Attendee: Shaohu, Appala, Hussein**

**Task:**

- Polish DD

**WENS lab 15 mins**

# **Acceptance Test Plan**

## **e-Climbing System**

**Prepared for:**  
**The Wellness Center Wall Climbing**  
**of**  
**South Dakota State University**



**Version 1.0**  
**Prepared By:**

**Nester Software. Inc**

**1400 8th St. Nester Center  
Brookings, SD 57006  
11/2016**

**Approved for public release; distribution is unlimited**

## Authoring & Approval

Name	Position & Title	Responsibility	Date
Shaohu Zhang	CEO Nester Software Inc.	Author	10/31/2016
Appala Chekuri	Software project manager Nester Software Inc.	Author	10/31/2016
Hussein Otudi	Systems Engineer Manager Nester Software Inc.	Author	10/31/2016
Justin Park	South Dakota State University	Approval	11/7/2016

## Reviewer

Name	Position & Title	Responsibility	Date
Shaohu Zhang	CEO Nester Software Inc.	Reviewer	10/31/2016

## Revision History

Date	Description	Revision	Editor
10/19/2016	Created the document	0	Shaohu
10/20/2016	Addition of introduction	0.1	Shaohu
10/22/2016	Addition of test principle	0.2	Shaohu
10/23/2016	Addition of hardware/software	0.3	Shaohu
10/23/2016	Addition of error handling policy	0.4	Shaohu
10/24/2016	Update of error handling policy	0.5	Shaohu
10/25/2016	Addition of project acceptance	0.6	Shaohu
10/26/2016	Update of hardware/software	0.7	Shaohu
10/27/2016	Update of introduction Addition of test sets	0.8	Shaohu Hussain
10/28/2016	Addition of test schedule Update of test sets	0.9	Hussain Shaohu
10/31/2016	Addition of unit test cases Update of test schedule Formatting Reviewer	1.0	Appala Hussain Shaohu

**This deliverable has two hard copies. One is for Dr. Shin from South Dakota State University, and another one is submitted to Mr. Park for approval. Distribution is unlimited for public release.**

**SOFTWARE ACCEPTANCE TEST PLAN  
FOR  
E-CLIMBING SYSTEM  
Prepared By:**

**Nester Software. Inc**

**1800 8th St. Nester Center**

**Brookings, SD 57006**

**11/2016**

**Team Information**

*Appala Chekuri*

---

**Software Project Manager** **Signature**

*Hussein Otudi*

---

**Systems Engineer Manager:** **Signature**

*Shaohu Zhang*

---

**Chief Executive Officer** **Signature**

# **CHAPTER 4 ACCEPTANCE TEST PLAN**

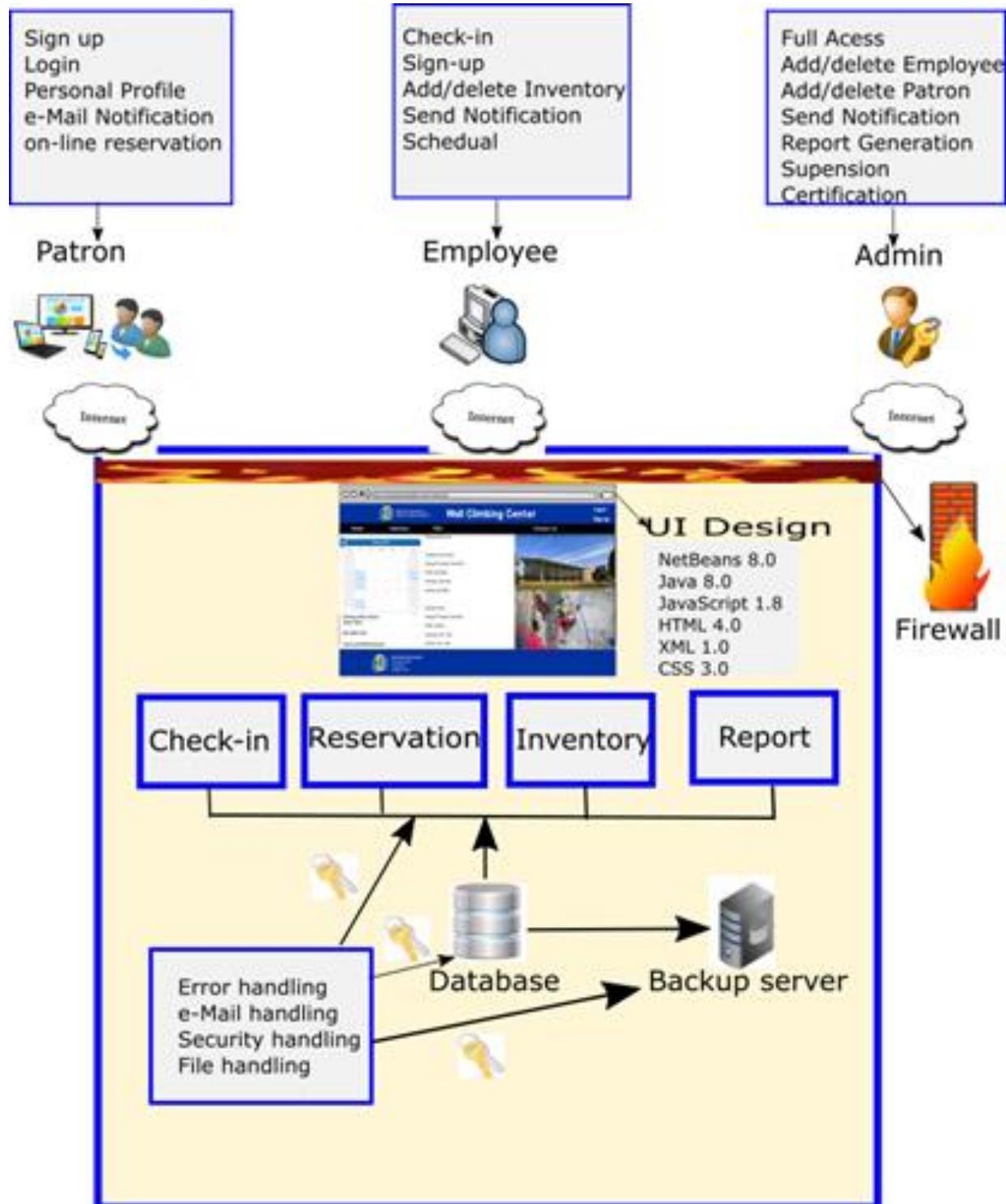
## **1. INTRODUCTION**

### **1.1 Purpose**

This document is to provide a clear statement with regard to the testing process for software and to outline who is responsible for which parts of that process. The process follows by the Software Design Document(SDD) which governed the design and implementation of the system. This document outlines four test cases of e-Climbing. Each test sets describes the detailed actions should be taken for testing and expected results.

Once the customer signs off on this document, \$20,000 (10% of contract payment) as listed in Proposal should be paid within 7 days.

## 1.2 System Overview



## 1.3 Test Principle

We use the following 8 processes to assess the quality and reliability of e-Climbing [2]:

### Principle 1: Definition

*To test a program is to try to make it fail.*

Its single goal is to uncover faults and defects by triggering failures. The definition also reminds us that testing unlike debugging does not deal with correcting faults, only checking for possible defects or bugs.

### Principle 2: Tests are no substitute for specifications

Specifications can serve to generate tests but defects happen because no one can completely think of some extreme case.

### Principle 3: Regression testing

Any failed execution must yield a test case, to remain a permanent part of the project's test suite. All failures occurring during development and testing should be covered.

### Principle 4: Manual and automatic test cases

An effective testing process must include both manually and automatically produced test cases. Manual tests are good at depth: They reflect developers' understanding of the problem domain and data structure. Automatic tests are good at breadth: They try many values including extremes that humans

### Principle 5: Testing a software early

It is imperative to start testing software as early as possible. This ensures that the defects can be captured and fixed within the stipulated time-frame, thereby allowing developers to deliver the software to the clients on time.

### Principle 6: Assessment criteria

A testing strategy's most important property is the number of faults it uncovers as a function of time.

### Principle 7: Clustering the defects

Defect clustering simply state that a small number of modules in an application contains maximum defects detected.

### Principle 8: Testing is dependent on context

This means that when you test a mobile application, it will be on different grounds than while testing a web application. Similarly, testing a Mac application will be different than testing an Android application and the likes.

## 1.4 Definitions, Acronyms, and Abbreviations

Table 1. Definitions, Acronyms, and Abbreviations

Terms	Definition, Acronym, Abbreviation
-------	-----------------------------------

ATP	Acceptance test plan
DFD	Design Flow Diagram
I/O	Input Output
Framework	Represents a collection of conceptual and technology mechanisms that provide a set of services and guidelines for applying a problem to particular domain.
POS	Point Of Sale
GPU	Graphics Processing Unit
WBS	Work Breakdown Structure
SDD	Software Design Document
EMS	Email Management System
DOB	Date Of Birth
DB	Database

## 1.5 Reference

[1].Software Design Document

[https://www.cs.drexel.edu/~dpn52/Therawii/acc\\_plan.pdf](https://www.cs.drexel.edu/~dpn52/Therawii/acc_plan.pdf)

[2]. Meyer, Bertrand. "Seven principles of software testing." *Computer* 41.8 (2008): 99-101.

[3]. Logging and Exception Handling in NetBeans

<https://bugzilla-attachments-35067.netbeans.org/bugzilla/attachment.cgi?id=28782>

## 1.6 Overview of Document

This document presents detailed information relating to the acceptance test plan for the e-Climbing software as shown in the following:

- Section 1 presents a brief overview of e-Climbing and the test principles driving its desire for software test.
- Section 2 lays out the hardware and software requirement for a proper test configuration of the e-Climbing system.
- Section 3 gives the test schedule with main functions.
- Sections 4 illustrates the test sets. Each test case gives the test sets, schedule and responsible person. There are 20 test cases in total.
- Sections 5 presents the errors handling policy.
- Sections 6 describes the individual case including test set with purpose, screen short, input and output. Once customer approves the test, the customer and developer will give signature.
- Section 7 is the meeting and review log.
- Section 8 gives the project acceptance. Once customer approves the test, both the customer and developer will give signature. The money should be paid as the schedule.

## 2. HARDWARE AND SOFTWARE FOR TESTING

This section describes the hardware and software for testing.

### 2.1 Hardware

Table 2 shows the testing configuration for the server(s) to test e-Climbing.

Table 2. Testing computer environment		
Manufacture	Lenovo	DELL
Model	T540P	Inspiron 5558
Processor	4th Gen Intel Core i5-4300M@2.4GHz 4	Inter(R)Core(TM)i3-4030U CPU 1.90GHz
Memory(RAM)	8 GB	6 GB
System Type	64 bit Operating System	64 bit Operating System
Operating System	Microsoft Windows 10	Windows 10 Home
Hard Disk	1TB Serial ATA Hard Drive	452 GB
Monitor	15.6 inch Widescreen Digital Plat Panel	Generic PnP Monitor
Optical Drive	16XDVD+/-RW Drive 12X RAM	P51F-001 DVD
Video Card	GeForce 6150SE nForce 430	Intel(R) HD Graphics Family
Keyboard & Mouse	USB Keyboard and Optical USB Mouse	Pen & touch
Wireless	Internal PCI 802.11g Wireless Network Card	Intel(R) Dual Band Wireless-AC3160
Anti Virus Software	McAfee SecurityCenter with anti-virus, anti-spyware, firewall	McAfee
Cords & Cables	Cat7 Ethernet cable, Power cords	Ethernet Realtek

### 2.2 Software

Table 3 shows the software configuration for the server(s) to test e-Climbing.

Table 3. Testing Server Software Configuration		
Items	Test 1 Software/System	Test 2 Software/System

Operating System	Windows 10	Windows 10
Antivirus	Norton Antivirus 2015, Symantec Endpoint Protection 11	Norton Antivirus 2015, Symantec Endpoint Protection 11
Web	Apache 2.2.11	Apache 2.2.11
Database	MySQL 5.0	MySQL 5.7
Java	Java 7.0	Java 8.0
HTML	HTML 5	HTML 5
CSS	CSS 3	CSS 3
Netbeans	Netbeans 8.0	Netbeans 8.1

### 3. TEST SCHEDULE

The test schedule is to ensure that the test plan meets the schedule.

**Table 4. Design priority table**

Test No	Test Item	Test description	Tester	Date
1	Check In	Manage the patron control with their access information into rock climbing center.	Shaohu	11/22/2016, 8:00 AM - 4:00 PM
2	Reservation	Used to reserve rooms for classes and meetings.	Hussain	11/24/2016, 8:00 AM - 4:00 PM
3	Inventory	Provides the list of items with their usage information.	Appala	11/26/2016, 8:00 AM - 4:00 PM
4	Report	Report will be generated by admin that will show the inventory tracking and list_serv in PDF format.	Shaohu	11/28/2016, 8:00 AM - 6:00 PM

## 4. TEST SETS

The test sets ensure the testing for individual set

**Table 5. Test Sets**

Test Items	Test method	Sets	Tester	Date
WBS NO 1.1.1 Register of patron	Verify registration form	1) Verify first name and last 2) Email verification	Shaohu	11/25/2016 8:00 AM - 4:00 PM
WBS NO 1.1.1.1 Waiver form	Verify Waiver form	Verify waiver form	Hussain	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.1.2 User validation	Verify email of user	Email verification	Apala	11/25/2016, 8:00 AM - 4:00 PM
WBS NO1.1.1.3 Payment method	Verify payment .	Verify if the patron have done payment or not	Shaohu	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.1.4 Update user information	Verify profile information	1) Update the phone number. 2) address	Hussain	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2 Log In	Verify USer Log-in	1) Verify the user_name 2) User _id	Apala	11/25/2016, 8:00 AM - 4:00 PM
WBS NO1.1.2.2 Employee Log-in	Verify Employee Log-in	Verify the employee name and password	Shaohu	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.2.1.1 Register class	Verify registration of patron to a class	Verify registration of patron to class	Hussain	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.2.1.2 EMS system	Verify Scheduling System for appointment, Class Timings and the notification remainder .	1) Verify time Availability of class	Appala	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.2.1.2.2 Send	Verify sending notifications by employee to the user	1) Verify notification set	Shaohu	11/25/2016, 8:00 AM - 4:00 PM

Notifications				
WBS NO 1.1.2.2.1.3 certification management	The certificate management will have the Track	1) Verify track certification 2) Verify update certification	Hussain	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.2.2.2	Verify whether the user/ patron has the access to the wall climbing center or not based on his Patron ID.	1) Verify Check-in verification	Aplala	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.2.2.5 Request admin for user suspension	Verify request of suspension by employee	1) Verify send request of suspension	Shaohu	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.3 Admin	Verify the adming Log_in.	1) Verify name 2) Verify password	Hussain	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.3.1 staff management	Verify staff management functionalities	1) Verify add new employee, and delete an employee		11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.3.2 admin for user suspension	Verify suspension and reason of suspension	1) Verify approval and rejection of suspension	Shaohu	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.3.3 Reservation	Verify The reservation	1) Verify past reservation schedule . 2) Availability of room for next meeting	Hussain	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.3.4 EMS system	Verify Scheduling System for appointment, Class Timings	1) Verify availability of time 2) Availability of class	Aplala	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.3.5 List Serv	Verify List_Sev functionality such as report of patron contact information	verify generating list of patron and their contact information	Shaohu	11/25/2016, 8:00 AM - 4:00 PM
WBS NO 1.1.2.3.6 Report generation	Verify report generation	1) verify generating inventory tracking report.	Hussain	11/25/2016, 8:00 AM - 4:00 PM

## 5. ERROR HANDLING POLICY

The following error handling policies are adhered to acceptance testing.

- **Exception handling in the application level.**

Table 5 gives the exception handling in the application level with 6 error levels. 1 has the most priority, 6 has the least priority.

**Table 5. Error level table**

Severity	Error Type	Responsible Person	Description	Example
1	SEVERE	CEO Shaohu	To handle an exception if it might be an important error (e.g. show it to the user):	<pre>try {     foo.doSomething(); } catch (IOException ioe) {     Logger.getAnonymousLogger().log(Level.SEVERE, "msg", ioe); }</pre>
2	CONFIG	Software Project Manager Appala	If it is not very important but should be sent to the log file.	<pre>try {     foo.doSomething(); } catch (IOException ioe) {     Logger.getAnonymousLogger().log(Level.CONFIG, "msg", ioe); }</pre>
3	INFO	Developer Hussain	If it is the normal outcome of a user action and there is no need to show stack traces to the user.	<pre>try {     foo.doSomething(); } catch (IOException ioe) {     Logger.getAnonymousLogger().log(Level.INFO, "msg", ioe); }</pre>
4	FINE	N/A	If the exception is not very important and there is no need to show or log file.	<pre>try {     foo.doSomething(); } catch (IOException ioe) {     Logger.getAnonymousLogger().log(Level.FINE, "msg", ioe); }</pre>
5	FINER	N/A	If the exception is less important and there is no need to show or log file.	<pre>try {     foo.doSomething(); } catch (IOException ioe) {     Logger.getAnonymousLogger().log(Level.FINER, "msg", ioe); }</pre>

6	FINEST	N/A	If the exception is the least important, and by default shall not be shown or logged at all .	<pre>try {     foo.doSomething(); } catch (IOException ioe) {     Logger.getAnonymousLogger().log(Level.FINEST,         "msg", ioe); }</pre>
---	--------	-----	---	--

- **Method-level error handling**

For the method-level errors, in particular, we'd like to take advantage of java.util.logging structured exception handling for dealing with errors .If potential errors are recoverable in the routine, use a combination of Try...Catch blocks as a retry mechanism for error handling.

## 6. INDIVIDUAL TEST CASES

The following Data Flow Diagrams(DFD), screenshot and input/output demonstrate the test case.

### 1.1.1 Registration of patron

WBS No. 1.1.1 registration	
Description	Allows user to enter the information and validate the information.
Validation	Valid email address
Input	First_name, last_name, email, confirm_email, password, confirm password , phone, DOB, Sex, and submit action
Output	Successful message with user unique id and his information.

#### TEST NO :1

**TEST PURPOSE:** To fill the registration form of new patron with their first and last name.

**INPUT:**

First\_name: Appala

Last\_name: Chekuri

**OUTPUT:** You can see the text filled with first name and the last name in the screen shot.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

#### TEST NO :2

**TEST PURPOSE:** After filling first name and the last name the user will fill out the Sex and Date of Birth in the following text fields.

**INPUT:**

First\_name: Appala

Last\_name: Chekuri

Date of Birth: 09/\*\*/\*\*\*\*

Sex: Male.

**OUTPUT:** You can see the Date of Birth and the sex of the patron filled in the text box.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

**TEST NO :3**

**TEST PURPOSE:** To fill the email address of the patron in order to get some notifications and to check the confirmation email has entered the same as the email entered before.

**INPUT:**

First\_name:Hussain

Last\_name: Otudi

Date of Birth: 09/\*\*/\*\*\*\*

Sex: Male.

Email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

Confirm email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

**OUTPUT:** You can see the email address of the patron entered in the text area where the confirmation email will be verified with the email entered before.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

**TEST NO :4**

**TEST PURPOSE:** To check the account duplication removal process. The email you entered in the field will be verified with the database for the duplicate. If it already exists then it will show the message that the email already exists and login .

**INPUT:**

First\_name:Hussain

Last\_name: Otudi

Date of Birth: 09/\*\*/\*\*\*\*

Sex: Male.

Email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

Confirm email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

**OUTPUT:** If the email already exists then the screen shows the message that the account with this email already exists. So login with the following email address.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

**TEST NO :5**

**TEST PURPOSE:** Enter the password into the text field and verify the password by confirm password option by asking the patron to re-enter it again.

**INPUT:**

First\_name:Hussain

Last\_name: Otudi

Date of Birth: 09/\*\*/\*\*\*\*

Sex: Male.

Email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

Confirm email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

Password: \*\*\*\*\*.

**OUTPUT:** If the password and the confirm password does not match then the system will ask the patron to enter the same password as you enter in the before field. Or show the message that the password does not match.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

**TEST NO :6**

**TEST PURPOSE:** This test purpose is to make sure that the password is at least 8 characters minimum

**INPUT:**

First\_name:Hussain

Last\_name: Otudi

Date of Birth: 09/\*\*/\*\*\*\*

Sex: Male.

Email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

Confirm email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

Password: \*\*\*\*\*.

**OUTPUT:** If the password is not 8 characters long then the system will ask the patron to enter the password which is at least 8 characters long.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

**TEST NO :7**

**TEST PURPOSE:** The patron will enter the phone number and click the submit button to enter the profile page.

**INPUT:**

First\_name:Hussain

Last\_name: Otudi

Date of Birth: 09/\*\*/\*\*\*\*

Sex: Male.

Email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

Confirm email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

Password: \*\*\*\*\*.

Phone number : 605-690-\*\*\*.

Click: Submit.

**OUTPUT:** Here the patron will enter the phone number and click the submit button so that he can enter the profile page.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

**TEST NO :8**

**TEST PURPOSE:** This case describes that after entering all the information correctly then the patron will successfully enter the profile page. It looks as the same in the screenshot.

**INPUT:**

First\_name:Hussain

Last\_name: Otudi

Date of Birth: 09/\*\*/\*\*\*\*

Sex: Male.

Email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

Confirm email: [narasimharaju386@gmail.com](mailto:narasimharaju386@gmail.com)

Password: \*\*\*\*\*.

Phone number : 605-690-\*\*\*.

Click: Submit.

**OUTPUT:** The patron profile page will be the output of this test case.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

### 1.1.2 Log In

WBS No. 1.1.2 login	
Description	This function for parton login, it will be displayed as html page asking user to enter username and password
Input	Here the input will be user_id and password
Output	The page will display for user include his/ her name on the top of homepage

#### TEST NO :1

**TEST PURPOSE:** In order to enter the profile page the patron, employee or the admin should login to his/her home profile by giving the username and the password in the login fields.

##### INPUT:

Patron\_ID:Raju23

Password:\*\*\*\*\*

**OUTPUT:** You will see the user name and the password typed in the fields on the login page.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

#### TEST NO :2

**TEST PURPOSE:** After typing the username and the password the user has to click the submit button in order to enter into the profile page.

##### INPUT:

Patron\_ID:Raju23

Password:\*\*\*\*\*

Click: Submit

**OUTPUT:** After entering username and the password the user will click the submit button to login to his page.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

#### TEST NO :3

**TEST PURPOSE:** when the user enters the username to login to his/her profile and when the username does not exist in the database then the system will show the message that the username does not exist.

##### INPUT:

Patron\_ID:Raju23

**OUTPUT:** You will see the error message that the username doesn't exist in the database and ask the user to enter the correct username.

**Customer signature** \_\_\_\_\_  
**Project team signature** \_\_\_\_\_  
**Date** \_\_\_\_\_  
**Comments** \_\_\_\_\_

**TEST NO :4**

**TEST PURPOSE:** After entering the username, password and clicking the Submit button and the password doesn't match then the user has to enter the correct password.

**INPUT:**

Patron\_ID:Raju23

Password:\*\*\*\*\*

Click: Submit

**OUTPUT:** If the user password is wrong then the system will show on the screen that the user has to enter the correct password in order to login to his page.

**Customer signature** \_\_\_\_\_  
**Project team signature** \_\_\_\_\_  
**Date** \_\_\_\_\_  
**Comments** \_\_\_\_\_

**TEST NO :5**

**TEST PURPOSE:** After Submitting the correct username and the password the user will enter the profile page,

**INPUT:**

Patron\_ID:Raju23

Password:\*\*\*\*\*

Click: Submit

**OUTPUT:** The output will be the profile page of the user.

**Customer signature** \_\_\_\_\_  
**Project team signature** \_\_\_\_\_  
**Date** \_\_\_\_\_  
**Comments** \_\_\_\_\_

### 1.1.2.2.2 check in management

WBS No. 1.1.2.2.2 Check In	
Description	The check- in system has very high priority for this system. Here the check In System will check whether the user/ patron has the access to the wall climbing center or not based on his Patron ID.
Validation	Patron's accessibility
Input	patron ID
Output	The output will be based on his Access to the rock climbing system. If it was accepted then he can enter the rock climbing center. If it was rejected then he will get the reason for reject.

Data Flow diagram for check in management:

**TEST NO :1**

**TEST PURPOSE:** will check whether the user/ patron has the access to the wall climbing center or not based on his Patron ID.

**Function Reference :WBS NO.1.1.1,WBs NO 1.1.2**

**INPUT:**Patron\_ID.

**OUTPUT:** accept or reject .

The patron ID will be entered in the textarea to check the access of the patron to enter the rock climbing center.

**Customer signature**\_\_\_\_\_

**Project team signature**\_\_\_\_\_

**Date**\_\_\_\_\_

**Comments**\_\_\_\_\_

**TEST NO :2**

**TEST PURPOSE:** Click submit in order to check the patron has access to the rock climbing center.

**Function Reference :WBS NO.1.1.1,WBs NO 1.1.2**

**INPUT:**Patron\_ID.

Click : submit.

**OUTPUT:** accept or reject .

**Customer signature**\_\_\_\_\_

**Project team signature**\_\_\_\_\_

**Date**\_\_\_\_\_

**Comments**\_\_\_\_\_

**TEST NO :3**

**TEST PURPOSE:** The patron will get the entrance denied if the payment has not done.

**Function Reference :WBS NO.1.1.1,WBs NO 1.1.2**

**INPUT:**Patron\_ID.

**OUTPUT:** accept or reject .

Accept will count one time for each check in, reject if the payment for membership is not done.

**Customer signature**\_\_\_\_\_

**Project team signature**\_\_\_\_\_

**Date**\_\_\_\_\_

**Comments**\_\_\_\_\_

**TEST NO :4**

**TEST PURPOSE:** Will show the access denied if the waiver form of the patron has not submitted.

**Function Reference :WBS NO.1.1.1,WBs NO 1.1.2**

**INPUT:**Patron\_ID.

**OUTPUT:** accept or reject .

Accept will count one time for each check in, reject will show message that ask to fill out waiver form

**Customer signature**\_\_\_\_\_

**Project team signature**\_\_\_\_\_

**Date**\_\_\_\_\_

**Comments**\_\_\_\_\_

**TEST NO :5**

**TEST PURPOSE:**will check whether the user/ patron has the access to the wall climbing center or not based on his Patron ID. And rejected if the patron is suspended from the rock climbing center by the administrator.

**Function Reference :WBS NO.1.1.1,WBs NO 1.1.2**

**INPUT:**Patron\_ID.

**OUTPUT:** accept or reject . reason of reject if the denied when the patron is suspended by the administrator.

**Customer signature**\_\_\_\_\_

**Project team signature**\_\_\_\_\_

**Date**\_\_\_\_\_

**Comments** \_\_\_\_\_

**TEST NO :6**

**TEST PURPOSE:** will check whether the user/ patron has the access to the wall climbing center or not based on his Patron ID. And give the access to the patron if the requirements are matched.

**Function Reference :WBS NO.1.1.1,WBs NO 1.1.2**

**INPUT:**Patron\_ID.

**OUTPUT:** accept or reject.

Accept will count one time for each check in.

**Customer signature** \_\_\_\_\_

**Project team signature** \_\_\_\_\_

**Date** \_\_\_\_\_

**Comments** \_\_\_\_\_

## 7. Log of Meetings

10/27 9-10:15 PM Thur	Spring meeting	WENS lab	75 mins
Attendee:	Shaohu, Appala, Hussein		
Task:			
• Split work for ATP			
• Set due			
10/28 9-9:15 PM Fri	Daily Scrum meeting	WENS lab	15 mins
Attendee:	Shaohu, Appala, Hussein		
Task:			
• Finish Unit test cases			
10/29 9-9:15 PM Sat	Daily Scrum meeting	WENS lab	15 mins
Attendee:	Shaohu, Appala, Hussein		
Task:			
• Finish test sets			
10/31 9-9:15 PM Sun	Daily Scrum meeting	WENS lab	15 mins
Attendee:	Shaohu, Appala, Hussein		
Task:			
• Polish ATP			

## 8. Project Acceptance

Once the document was signed off, no more features can be requested. The cost will have to be renegotiated if the customer has an addition feature . By signing this document everyone agrees that all requirements have been met. \$20,000 (10% of contract payment) as listed in Proposal should be paid within 7 days.

**Customer:**

\_\_\_\_\_  
\_\_\_\_\_  
**(Signature)**  
**(Printed Name)**

**Title:** \_\_\_\_\_

**Date:** \_\_\_\_\_

**Developer:**

\_\_\_\_\_  
\_\_\_\_\_  
**(Signature)**  
**(Printed Name)**

**Title:** \_\_\_\_\_  
**Date:** \_\_\_\_\_

# **System Test Plan**

## **e-Climbing System**

**Prepared For:**  
The Wellness Center Wall Climbing  
of  
South Dakota State University



**Version 2.0**

**Prepared By:**  
**Nester Software. Inc**

**1400 8th St. Nester Center  
Brookings, SD 57006  
12/2016**

**Approved for public release; distribution is unlimited**

## Authoring & Approval

Name	Position & Title	Responsibility	Date
Shaohu Zhang	CEO Nester Software Inc.	Author	12/03/2016
Appala Chekuri	Software project manager Nester Software Inc.	Author	12/03/2016
Hussein Otudi	Systems Engineer Manager Nester Software Inc.	Author	12/03/2016
Justin Park	South Dakota State University	Approval	12/03/2016

## Reviewer

Name	Position & Title	Responsibility	Date
Shaohu Zhang	CEO Nester Software Inc.	Reviewer	12/03/2016

## Revision History

Date	Description	Revision	Editor
11/24/2016	Created the document	0	Shaohu
11/24/2016	Addition of introduction	0.1	Shaohu
11/25/2016	Addition of test items	0.2	Shaohu
11/25/2016	Addition of hardware/software	0.3	Shaohu
11/26/2016	Addition of Transaction flow testing	0.4	Appala Hussein
11/26/2016	Addition of Non-functional requirements testing	0.5	Shaohu
11/27/2016	Update of Transaction flow testing	0.6	Appala Hussein
11/27/2016	Update of hardware/software	0.7	Shaohu
11/27/2016	Update of introduction	0.8	Shaohu Hussain

11/28/2016	Addition of unit testing	0.9	Hussain Shaohu
11/29/2016	Formatting Reviewer	1.0	Shaohu
12/03/2016	Polish	2.0	Shaohu Appala Hussain

**This deliverable has two hard copies. One is for Dr. Shin from South Dakota State University, and another one is submitted to Mr. Park for approval. Distribution is unlimited for public release.**

**SOFTWARE SYSTEM TEST PLAN**  
**FOR**  
**E-CLIMBING SYSTEM**

Prepared by:  
**Nester Software. Inc**  
1800 8th St. Nester Center  
Brookings, SD 57006  
11/2016

**Team Information**

*Appala Chekuri*

---

**Software Project Manager**

**Signature**

*Hussein Otudi*

---

**Systems Engineer Manager:**

**Signature**

*Shaohu Zhang*

---

**Chief Executive Officer**

**Signature**

# CHAPTER 5 SYSTEM TEST PLAN

## 1. INTRODUCTION

### 1.1 Purpose

This document is to provide a clear statement with regard to the testing plan for the developed software system against the software requirements as defined in the Requirements Documentation. The purpose for these system tests is to make sure that the software system developed during the e-Climbing System project complies with the definition of the software requirements specified by the Nester Software Company. This document describes the approach to testing that the Nester team will use to verify that the application meets the established requirements of e-Climbing prior to release. Together, this plan and the following tests define the testing to be performed on the e-Climbing System by Nester team members and representatives from Nester Software.

Once the customer signs off on this document, \$20,000 (10% of contract payment) as listed in Proposal should be paid within 7 days.

### 1.2 Scope of Product

The software product to be produced shall hereby be referred to as the e-Climbing System and also referred to as “the system.” The e-Climbing System will handle daily patron management operations and allow the users to manage inventory and generate report. To achieve this the system must be able to handle two types of users.

The first type of user is the Staff users. The system will provide the Staff users with the ability to do check-in, registration, reservation, update user's information and inventory management. The second type of user will be the administrators of the system. The system will grant them access to add/update Staff information, update schedule and generate report. The system will also grant administrators the access to delete any patron and Staff.

### 1.3 Definitions, Acronyms, and Abbreviations

Table 1. Definitions, Acronyms, and Abbreviations	
Terms	Definition, Acronym, Abbreviation
ATP	Acceptance test plan

DFD	Design Flow Diagram
I/O	Input Output
GPU	Graphics Processing Unit
FWBS	Functional Work Breakdown Structure
STP	System Test Plan
EMS	Email Management System
DOB	Date Of Birth
NTT	Nester Testing Team
BVA	Boundary Value Analysis
RTM	Requirement Traceability Matrix

## 1.4 Reference

[1]. Software Design Document

[https://www.cs.drexel.edu/~dpn52/TheRawii/acc\\_plan.pdf](https://www.cs.drexel.edu/~dpn52/TheRawii/acc_plan.pdf)

[2]. Meyer, Bertrand. "Seven principles of software testing." *Computer* 41.8 (2008): 99-101.

## 1.5 Overview of Document

This document presents detailed information relating to the STP for the e-Climbing software as shown in the following:

- Section 1 presents a brief overview of e-Climbing and the purpose driving its desire for system test plan.
- Section 2 outlines the test items to be tested or not.
- Section 3 lays out the hardware and software requirement for a proper testing of the e-Climbing system.
- Section 4 describes the detailed information about transaction flow testing. The black box tests are applied.
- Sections 5 illustrates the non-functional requirement testing in terms with compatibility test, regression test with policy and backup test.
- Sections 6 presents 3 unit testing case using white box testing approach.

## 2. TEST ITEMS

Test	FWBS #	Function	Description	Test/Not	Passed/Not
------	--------	----------	-------------	----------	------------

#				Tested	Passed
1	1.1	Sign in	admin/staff sign the user and password, if he/she has already signed, log in, otherwise ask for registration	Yes	
2	1.1.1	Registration	New user register a new account	Yes	
3	1.1.2	Log in	admin/staff sign the user and password,	Yes	
4	1.1.3	Support desk info	Show the contact information of software admin	No	
5	1.1.1.1	Waiver form	If the patron is not signed off, ask sign waiver form.	No	
6	1.1.1.2	Validate user	Validate user membership	Yes	
7	1.2	Inventory	Equipment item management	Yes	
8	1.2.1	Equipment data	Show model , quality, expiration et.cl.	No	
9	1.2.2	Certification level	Including Level 1,2 and 3	Yes	
10	1.1.2.3	Administrator	Admin component	Yes	
11	1.1.2.3.1	Reservation	User can make a schedule	Yes	
12	1.1.2.3.3	Approve suspend patron	Suspend patron to access climbing wall	Yes	
13	1.2.3.4	Manage staff	Add, delete/ update staff information	Yes	
14	1.1.2.3.5	Generate report	Produce report by daily, weekly et.al.	Yes	
15	1.1.2.2.2.2	Check in	The staff check in the patron	Yes	
16	1.1.2.2.1.1	Register class	Register class for patron	Yes	

### 3. SOFTWARE TESTING APPROACH

#### 3.1 Software testing team

The e-Climbing system testing team belongs to the Nester Testing Team(NTT) in charge of Shaohu Zhang. The NIT is separated from the e-Climbing developer team. The following table shows testing team members and testing responsibility for each NIT member.

Name	Responsibility			
	Develop ATP	Report progress	Testing	Review and approve
Shaohu	√	√		√
Hussain	√		√	
Appala	√		√	

### 3.2 Software testing process

The testing process involves different types of black box testing and white box testing according the requirement of the module.

- We shall perform tests like positive tests, negative test, requirement based test, state based and Cause and Effect testing for all of the modules. These test verify that the system is working according to the specified requirement.
- We shall also perform Boundary Value Analysis and Equivalence Partition to see that the application is working correctly in the boundary regions of some modules. Through the use of such type of test we can verify that the system is designed strong enough to perform its tasks effectively.
- Furthermore, other tests like compatibility will be done to show which platform and environment is supported by the application and hence find the minimum requirement that is needed to run the website.
- The backup and recovery test will also be performed to see how stable and how efficient the system can perform.

### 3.3 Hardware and software requirements

Two laptops are used to test e-Climbing system.

#### 3.3.1 Hardware requirements

Below table shows the hardware requirements for the testing.

Hardware	Test 1	Test 2
Manufacture	Lenovo	DELL
Model	T540P	Inspiron 5558
Processor	4th Gen Intel Core i5-4300M@2.4GHz 4	Inter(R)Core(TM)i3-4030U CPU 1.90GHz
Memory(RAM)	8 GB	6 GB

System Type	64 bit Operating System	64 bit Operating System
Operating System	Microsoft Windows 10	Windows 10 Home
Hard Disk	1TB Serial ATA Hard Drive	452 GB
Monitor	15.6 inch Widescreen Digital Plat Panel	Generic PnP Monitor
Optical Drive	16XDVD+/-RW Drive 12X RAM	P51F-001 DVD
Video Card	GeForce 6150SE nForce 430	Intel(R) HD Graphics Family
Keyboard & Mouse	USB Keyboard and Optical USB Mouse	Pen & touch
Wireless	Internal PCI 802.11g Wireless Network Card	Intel(R) Dual Band Wireless-AC3160
Anti Virus Software	McAfee SecurityCenter with anti-virus, anti-spyware, firewall	McAfee
Cords & Cables	Cat7 Ethernet cable, Power cords	Ethernet Realtek

### 3.3.2 Software requirements

Below table shows the software requirements for the testing.

Software	Test 1	Test 2
Operating System	Windows 10	Windows 7
Antivirus	Norton Antivirus 2015, Symantec Endpoint Protection 11	Norton Antivirus 2015, Symantec Endpoint Protection 11
Web	Apache 2.2.11	Apache 2.2.11
Database	MySQL 5.0	MySQL 5.7
Java	Java 7.0	Java 8.1
HTML	HTML 5	HTML 5
CSS	CSS 3	CSS 3
Netbeans	Netbeans 8.0	Netbeans 8.1

## 4. TRANSACTION FLOW TESTING

### 4.1 Test1. Registration of patron

#### 4.1.1 Sprint & FWBS

FWBS Number: 1.1.1

Sprint Number: S1

#### 4.1.2 Technique 1: Functionality and Procedure

Function	What you need to know?
<b>Firstname</b>	<p>Enter first name of the patron for registration</p> <ol style="list-style-type: none"><li>If the first name of the user is less than 4 characters, then it displays a message that first name should not be less than 4 characters.</li><li>It allows successfully the patron to enter the last name if the first name is filled and the more than 4 characters.</li></ol> <p><b>Expected output:</b> See screenshot 4.1.</p>
<b>Lastname</b>	<p>Enter last name of the patron for registration.</p> <ol style="list-style-type: none"><li>If the last name of the user is less than 4 characters, then it displays a message that last name should not be less than 4 characters.</li><li>It allows successfully the patron to enter the password if the last name is filled and the more than 4 characters.</li></ol> <p><b>Expected output:</b> See screenshot 4.2</p>
<b>Username</b>	<p>Enter the username of the patron for registration.</p> <p>Check if the username match with the requirement</p> <ol style="list-style-type: none"><li>The username of the patron should not be less than 8 characters. If it was less than 8 characters, then it displays a message to enter at least 8 characters.</li><li>Once the patron enters the 8-character username then the patron can go to next field to fill the password.</li></ol> <p><b>Expected output:</b> see screenshot 4.3</p>
<b>Password</b>	<p>Check if the password filled in the field is according to the requirement or not</p> <ol style="list-style-type: none"><li>If the password is typed less than 7 characters, then it will show the message to enter at least 7 characters.</li><li>When the user enters the more than 7 character then the screen control can change to confirm password.</li></ol>

	<p>3. If the password is more than 15 characters, it won't accept to submit the registration.</p> <p><b>Expected output:</b> see screenshot 4.4</p>
<b>Confirm_Password</b>	<p>Check if the confirm password is matching the password or not</p> <ol style="list-style-type: none"> <li>1. If the password matches, then the patron can go to email field and fill it for registration.</li> <li>2. If the password doesn't match with the password, then the error message show that the confirm password should match the password.</li> </ol> <p><b>Expected output:</b> See screenshot 4.5</p>
<b>Email</b>	<p>Check the validated email is entered in the field or not</p> <ol style="list-style-type: none"> <li>1. If the email is not valid or not according to the format of the email addressing standards, then the registration will not be accepted.</li> <li>2. The email is entered according to the valid standard then you can get the registration submitted.</li> </ol> <p><b>Expected output:</b> See the screenshot 4.6</p>
<b>Phone</b>	<p>The phone number of the patron will be 10-character long</p> <ol style="list-style-type: none"> <li>1. If less than 10 characters, the it will show the error that the phone number should be 10-character long.</li> <li>2. If more than 10 characters, then it will show the error that the phone number should be 10-character long.</li> <li>3. If exactly 10 characters long then it will be accepted to be submit the registration.</li> </ol> <p><b>Expected output:</b> See screenshot 4.7.</p>
<b>Semester</b>	Here the patron has to select the semester which he/she want to register for the rock rock climbing from the dropdown list.
<b>Sex</b>	Here the patron will have the drop down list to select the sex of the patron .
<b>DOB</b>	<p>The date of birth of the patron will be typed here in the format(yyyy-mm-dd) mentioned on the screen.</p> <ol style="list-style-type: none"> <li>1. If the patron enters the wrong format it will not accept the patron to submit other details for submit.</li> </ol> <p><b>Expected output:</b> see screenshot 4.8</p>
<b>Inputs</b>	See screenshot for input and output of registration.
<b>Expected Output</b>	See screenshot for input and output information

#### 4.1.3 Technique 2: Causes and effect

#### **4.1.3.1 causes:**

1. If first name less than 4 characters
2. If last name less than 4 characters
3. If username less than 4 characters
4. If password less than 7 characters
5. If confirm password less than 7 characters
6. If email address validation check
7. Phone number not equal to 10 numbers
8. Date of birth not in format.
9. All data entered correctly

#### **4.1.3.2 Effect**

101. Return back to register page
102. Sign in to valid home page.

#### **4.1.4 Technique 2: Negative Testing**

Subfunction	Input	Message	Output	
<b>Firstname</b>	5838@	Incorrect	Enter valid data	
<b>Lastname</b>	24_+%	Incorrect		
<b>Username</b>	!2#udj	Incorrect		
<b>Password</b>	11222	Password doesn't match		
<b>Confirm_Password</b>	3322222			
<b>Email</b>	fjdjdjj@	Incorrect		
<b>Phone</b>		Phone Field is required		
<b>DOB</b>	12-1333-11	Incorrect format		

#### 4.1.5 Boundary Value Analysis & Equivalence Partitioning

Test no	First name length	Last name length	Password length	Confirm password length	Phone number length
1	0	Not provided	0	0	0
2	1	Not provided	1	1	1
3	aa	Not provided	aa	aa	123
4	aaa	Not provided	password<7	password<7	Phone number <10
5	aaaa	Not provided	password=7	password=7	Phone number =10
6	First name>4	Not provided	password>7	password>7	Phone number >10
7	0	0	0	0	0
8	1	1	1	1	1
9	aa	aa	aa	aa	123
10	aaa	aaa	password<7	password<7	Phone number <10
11	aaaa	aaaa	password=7	password=7	Phone number =10
12	First name>4	Last name>4	password>7	password>7	Phone number >10
13	0	0	0	0	0
14	1	1	1	1	1
15	aa	aa	aa	aa	123
16	aaa	aaa	password<7	password<7	Phone number <10
17	aaaa	aaaa	password=7	password=7	Phone number =10
18	First name>4	Last name>4	password>7	password>7	Phone number >10

19	0	0	0	0	0
20	1	1	1	1	1
21	aa	aa	aa	aa	123
22	aaa	aaa	password<7	password<7	Phone number <10
23	aaaa	aaaa	password=7	password=7	Phone number =10
24	First name>4	Last name>4	password>7	password>7	Phone number >10
25	0	0	0	0	0
26	1	1	1	1	1
27	aa	aa	aa	aa	123
28	aaa	aaa	password<7	password<7	Phone number <10
29	aaaa	aaaa	password=7	password=7	Phone number =10
30	First name>4	Last name>4	password>7	password>7	Phone number >10

#### 4.1.6 Merged Test Cases

Test Number	Label	Input	Expected Output	Test cases 1	Test cases 2
TC1	Firstname	Hussain	Successfull y added new patron See screenshot 1		
	Lastname	Otudi			
	Username	Huss123			
	Password	Huss123			
	Confirm_Password	Huss123			

	<b>Email</b>	Otudi.2015@gmail.com			
	<b>Phone</b>	605-592-0069			
	<b>Semester</b>	Fall			
	<b>Sex</b>	Male			
	<b>DOB</b>	1989-04-14			
TC2	<b>Firstname</b>	5838@	Enter valid data See screenshot 2		
	<b>Lastname</b>	24_+%			
	<b>Username</b>	!2#udj			
	<b>Password</b>	11222			
	<b>Confirm_Password</b>	3322222			
	<b>Email</b>	fjdjdjj@			
	<b>Phone</b>				
	<b>DOB</b>	12-1333-11			
	0	Not provided	0	0	0
	1	Not provided	1	1	1
	aa	Not provided	aa	aa	123
	aaa	Not provided	password<7	password<7	Phone number <10
	aaaa	Not providedNot provided	password=7	password=7	Phone number =10

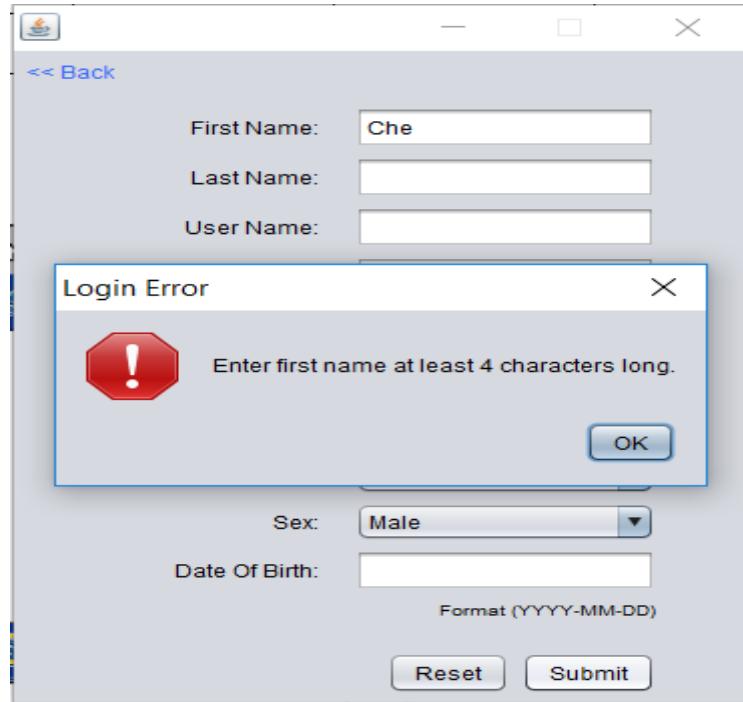
	First name>4		password>7	password>7	Phone number >10
	0	0	0	0	0
	1	1	1	1	1
	aa	aa	aa	aa	123
	aaa	aaa	password<7	password<7	Phone number <10
	aaaa	aaaa	password=7	password=7	Phone number =10
	First name>4	Last name>4	password>7	password>7	Phone number >10
	0	0	0	0	0
	1	1	1	1	1
	aa	aa	aa	aa	123
	aaa	aaa	password<7	password<7	Phone number <10
	aaaa	aaaa	password=7	password=7	Phone number =10
	First name>4	Last name>4	password>7	password>7	Phone number >10
	0	0	0	0	0
	1	1	1	1	1
	aa	aa	aa	aa	123

	aaa	aaa	password<7	password<7	Phone number <10
	aaaa	aaaa	password=7	password=7	Phone number =10
	First name>4	Last name>4	password>7	password>7	Phone number >10
	0	0	0	0	0
	1	1	1	1	1
	aa	aa	aa	aa	123
	aaa	aaa	password<7	password<7	Phone number <10
	aaaa	aaaa	password=7	password=7	Phone number =10
	First name>4	Last name>4	password>7	password>7	Phone number >10

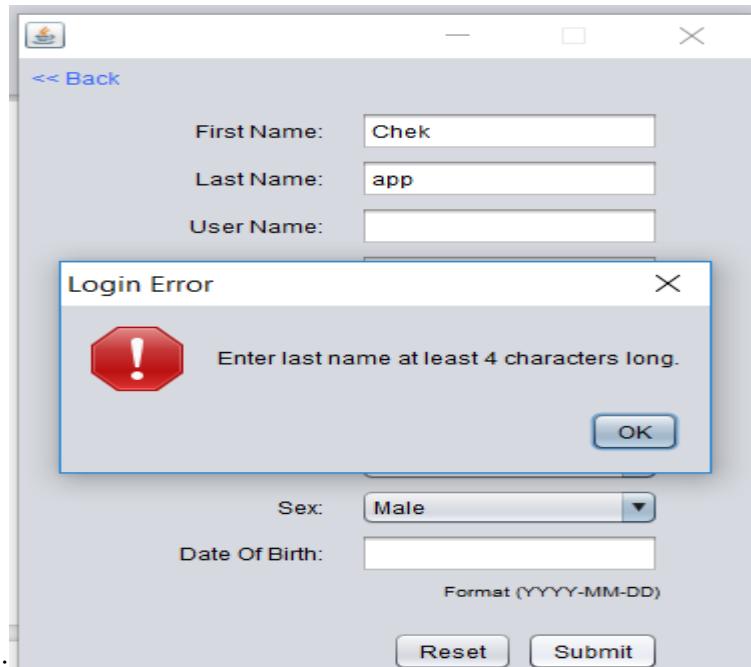
#### 4.1.7 Requirement Traceability Matrix

FWBS Number	Description	Priority (H,M,L))	Test Condition	Test Case_ID	Phase of testing
FWBS :1.1.2.2.1.2.2	Using valid information	H	User get registered	TC1	integration function
FWBS :1.1.2.2.1.2.2	Using invalid information	H	User does not get registered	TC2	Integration Function
FWBS :1.1.2.2.1.2.2	Send email to user for successful registration	M	When User get registered he/she receive an email with patron_id	TC1	Integration function

**Registration:**



Screenshot 4.1



Screenshot 4.2

The screenshot shows a user registration form. The User Name field contains "naras", which is less than 8 characters long. An error message box titled "Login Error" displays the message "Enter username at least 8 characters long." with an exclamation mark icon.

First Name: Chek  
Last Name: appa  
User Name: naras

Login Error

Enter username at least 8 characters long.

OK

Sex: Male  
Date Of Birth:   
Format (YYYY-MM-DD)  
Reset Submit

Screenshot 4.3

The screenshot shows a user registration form. The Password field contains "\*\*\*\*", which is less than 7 characters long. An error message box titled "Login Error" displays the message "Enter password with 7 -15 characters long." with an exclamation mark icon.

First Name: Chek  
Last Name: appa  
User Name: narasimha  
Password: \*\*\*\*  
Confirm Password:

Login Error

Enter password with 7 -15 characters long.

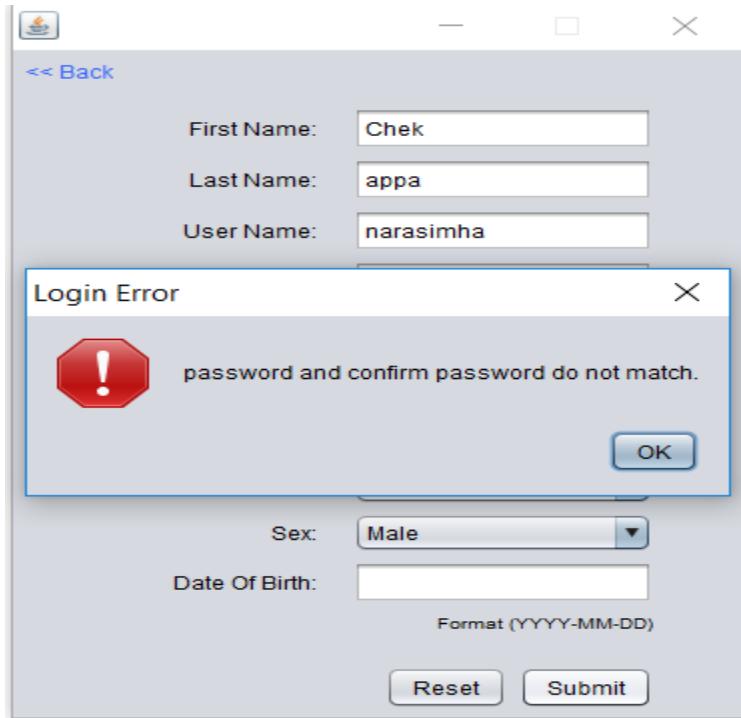
OK

Format (YYYY-MM-DD)  
Reset Submit

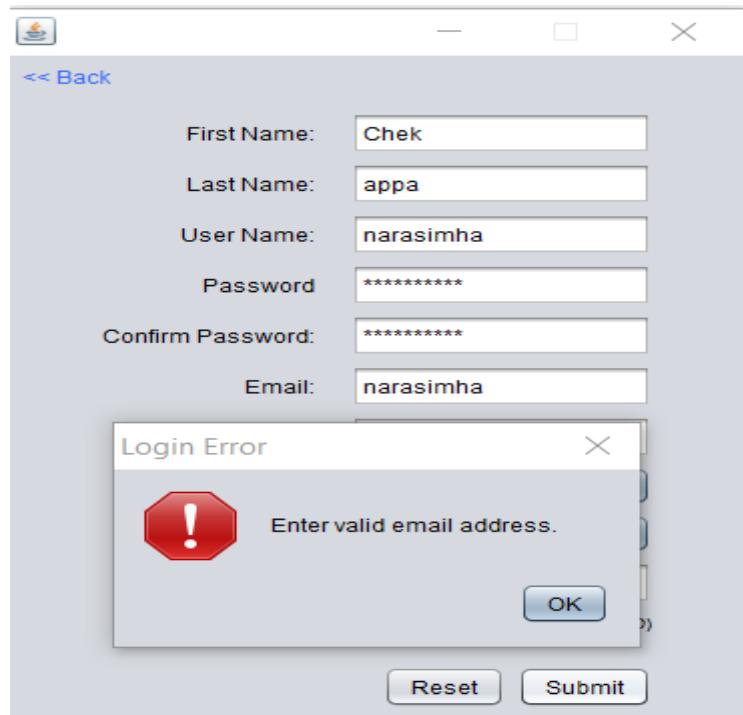
Screenshot 4.4



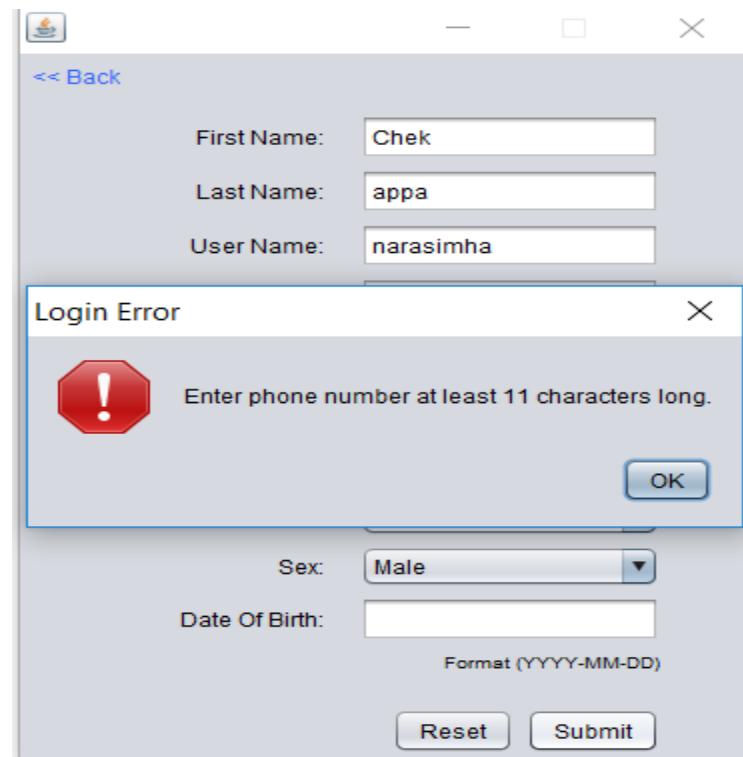
Screenshot 4.5



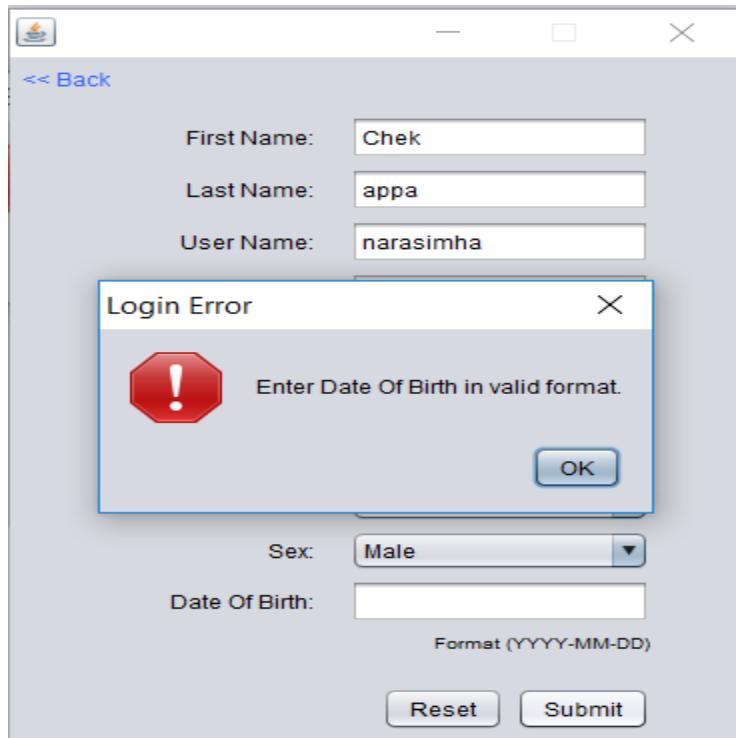
Screenshot 4.5.1



Screenshot 4.6



Screenshot 4.7



Screenshot 4.8

## 4.2 Test 2. Log In

### 4.2.1.Sprint & FWBS

FWBS Number : 1.1.2

FWBS Number Patron : 1.1.2.1

FWBS Number Employee : 1.1.2.2

FWBS Number Admin : 1.1.2.3

Sprint Number :S2

### 4.2.1.2 Functionality and Procedure

Function	What we need to know
Username	Checking if the username is matched with requirement. And is present in the database or not
Password	Checking if the password is matched with requirement.

Action	Enter username and password and click “Submit” button.
Inputs	Username , Password and submit
Expected Output	<ol style="list-style-type: none"> <li>1. Successful login</li> <li>2. Redirect page to user profile</li> <li>3. Unsuccessful login</li> <li>4. <ul style="list-style-type: none"> <li>• Error message for Invalid</li> <li>• username or password (Refer</li> <li>• Error message for excessing</li> <li>• more than 3 times to log in</li> </ul> </li> <li>5. c. Error message when user is not registered.</li> </ol>

#### 4.2.2 Technique 1: Positive Testing

Subfunction	Input	Message	Output
Username	Huss1234	N/A	Redirect to home page
Password	12345	N/A	

#### 4.2.3 Technique 2: Negative Testing

Subfunction	Input	Message	Output
Username	@385884%	The email or password you entered is incorrect	<b>Remain in login page</b>
Password	@#%t67		

## 4.2.4 Technique 3: Decision Table and Cause-Effective Graph

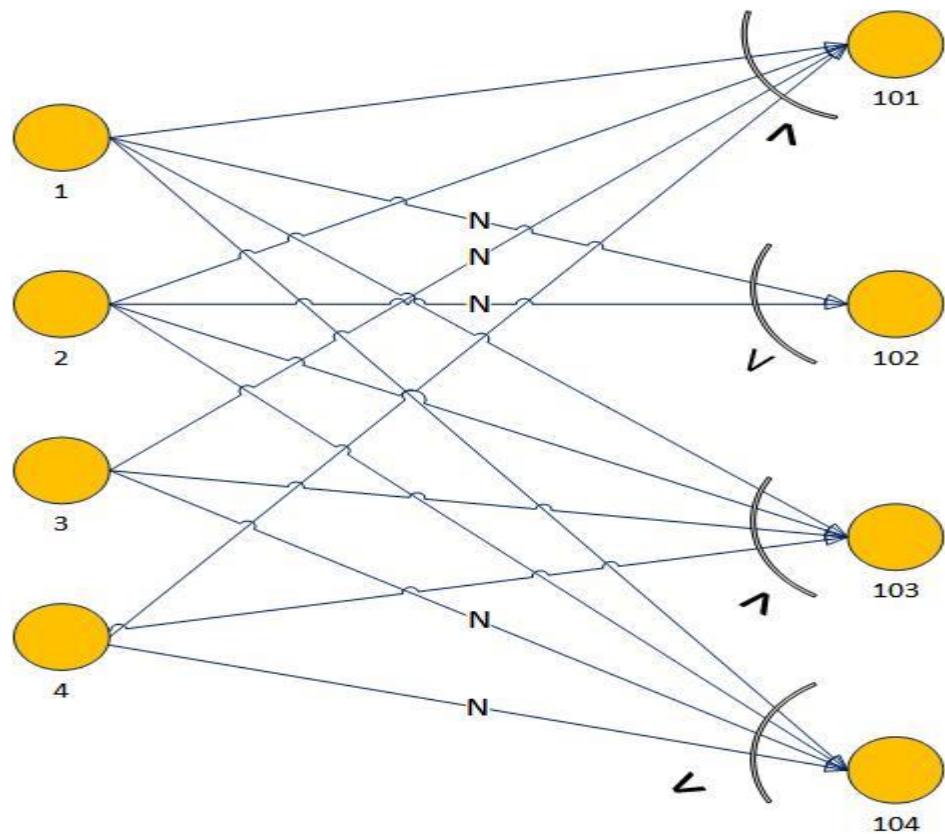
### 4.2.4.1 Causes

1.  $0 < \text{Username Length} \leq 15$
2.  $0 < \text{Password Length} \leq 15$
3. Username and Password are in Database
4. User is suspended

### 4.2.4.2 Effects

- 101: Redirect to main page.
- 102: Error 1 (Username and password don't match)
- 103: Error 2 (User is not in the database)
- 104: Redirect to main page with storing sign up info in the catch.

### 4.2.4.3 Cause-Effect Graph



#### 4.2.4.4 Cause-Effect Table

Causes	P1	P2	P3	P4
1	1	0	0	0
2	0	1	0	0
3	1	1	1	0
4	1	1	1	1
Effects				
101	1	1	1	0
102	0	0		
103	1	1	1	1
104	1	1	1	0

#### 4.2.5 Equivalence Partitioning and BVA

Test Number	Username	Password	Length of Username	Length of Password
1	Not provide	Not provide	0	0
2	Not provide	B		1
3	Not provide	Rajjj		1<<14
4	Not provide	RrHussainSahhoi		14
5	Not provide	RrHussainSahhoie		15
6	Not provide	RrHussainSahhoiew		16
7	R		1	0
8	R	ch		1
9	R	checori		1 <<14
10	R	RrHussainSahho1		14
11	R	RrHussainSahho12		15
12	R	RrHussainSahhoie1		16
13	raju		1 <<14	0
14	raju	ch		1
15	raju	checori		1 <<14
16	raju	RrHussainSahhoi		14
17	raju	RrHussainSahhoi2		15
18	raju	RrHussainSahhoie12		16

19	rajusds	Not provide	14	0
20	rajusds	ch		1
21	rajusdsds	checori		1< <14
22	rajudfds	RrHussainSahho1		14
23	rajufdfd	RrHussainSahho12		15
24	rajutre	RrHussainSahhoie1		16
25	RrHussainSahhod	Not provide	15	0
26	RrHussainSahhod	ch		1
27	RrHussainSahhod	checori		1 < <14
28	RrHussainSahhod	RrHussainSahho1		14
29	RrHussainSahhod	RrHussainSahho12		15
30	RrHussainSahhod	RrHussainSahhoie1		16
31	RrHussainSahhoieh	Not provide	16	<b>0</b>
32	RrHussainSahhoieh	ch		1
33	RrHussainSahhoieh	checori		1< <14
34	RrHussainSahhoieh	RrHussainSahho1		14
35	RrHussainSahhoieh	RrHussainSahho12		15
36	RrHussainSahhoieh	RrHussainSahhoie1		16

#### 4.2.6 Merge Test Cases

Test Number	Username	Password	Expected Output
TC1	Raju1234	R12345678	See screenshot
TC2	@raju1234	@@@@@@	See screenshot
TC3	&12husain	Not provide	See screenshot
TC4	Not provide	B	
TC5	Not provide	Rajjj	
TC6	Not provide	RrHussainSahhoi	
TC7	Not provide	RrHussainSahhoie	
TC8	Not provide	RrHussainSahhoiew	
TC9	R		
TC10	R	ch	
TC11	R	checori	See screenshot
TC12	R	RrHussainSahho1	
TC13	R	RrHussainSahho12	
TC14	R	RrHussainSahhoie1	
TC15	raju		See screenshot
TC16	raju	ch	See screenshot
TC17	raju	checori	
TC18	raju	RrHussainSahhoi	See screenshot

TC19	raju	RrHussainSahhoi2	
TC20	raju	RrHussainSahhoie12	See screenshot
TC21	rajusds	Not provide	
TC22	rajusds	ch	
TC23	rajusdsds	checori	
TC24	rajudfds	RrHussainSahho1	See screenshot
TC25	rajufdfd	RrHussainSahho12	
TC26	rajutre	RrHussainSahhoie1	See screenshot
TC27	RrHussainSahhod	Not provide	
TC28	RrHussainSahhod	ch	
TC29	RrHussainSahhod	checori	
TC30	RrHussainSahhod	RrHussainSahho1	See screenshot
TC31	RrHussainSahhod	RrHussainSahho12	
TC32	RrHussainSahhod	RrHussainSahhoie1	
TC33	RrHussainSahhoieh	Not provide	
TC34	RrHussainSahhoieh	ch	
TC35	RrHussainSahhoieh	checori	
TC36	RrHussainSahhoieh	RrHussainSahho1	

#### 4.2.7 Requirement Traceability Matrix

FWBS Number	Description	Priority (H,M,L))	Test Condition	Test Case_ID	Phase Of testing
1.1.2.1	Using valid username to login	H	Number to attempt to login <15	TC1, TC2 and TC3, and all test cases from TC9 to TC36	Integration function
1.1.2.1	Using invalid username to login	H	Number to attempt to login <15	From TC4 to TC8	Integration function
1.1.2.1	Using valid username and no password is Provide	H	Number to attempt to login <15	TC15 TC21 TC33	Integration
1.1.2.1	Using valid username and invalid password	H	Please Enter valid password	TC2,TC24,25,26,TC33,T C34,TC35,TC36	Integration

### 4.3 Test 3. Check-in management

#### 4.3.1 Sprint & FWBS

FWBS Number: 1.1.2.2.2

Sprint Number: S3

#### 4.3.2 Functionality and Procedure

##### 4.3.1.2.1 Registered User

FWBS:1.1.2.2.2	
Function Description	What we need to know
Patron ID	The patron has to enter the patron ID In order to enter into the rock climbing center.  1. If the patron is not registered then he can register at the rock

	<p>climbing center.</p> <ol style="list-style-type: none"> <li>2. If the patron has denied because of the suspension form the admin due to overcoming the policies. Then the information is shown on the page.</li> <li>3. If the patron has entered wrong patron ID           <ol style="list-style-type: none"> <li>a. Patron ID less than 6 digit.</li> <li>b. Patron ID more than 6 digit.</li> <li>c. Wrong patron ID.</li> </ol> </li> </ol>
<b>Inputs</b>	The patron has entered into the text field as Input.
<b>Expected Output</b>	See the screenshot for output.

#### 4.3.1.2.2 Demographic user

FWBS:1.1.2.2.2.2	
Function Description	Procedure
<b>Demographic Patron</b>	The demographic patron will not have the patron ID as the registered patron. SO enter following details and submit the waiver form.
<b>Patron name</b>	The demographic user has to enter the username to enter rock climbing center.
<b>Waiver form</b>	The demographic user has to fill the waiver form at the rock climbing center.
<b>Inputs</b>	The patron has entered into the text field as Input.
<b>Expected Output</b>	See the screenshot for output.

#### 4.3.3 Technique 1: Boundary value analysis

Subfunction	Input length	Output
<b>Patron ID</b>	Not provided	Show check IN Details
	1	

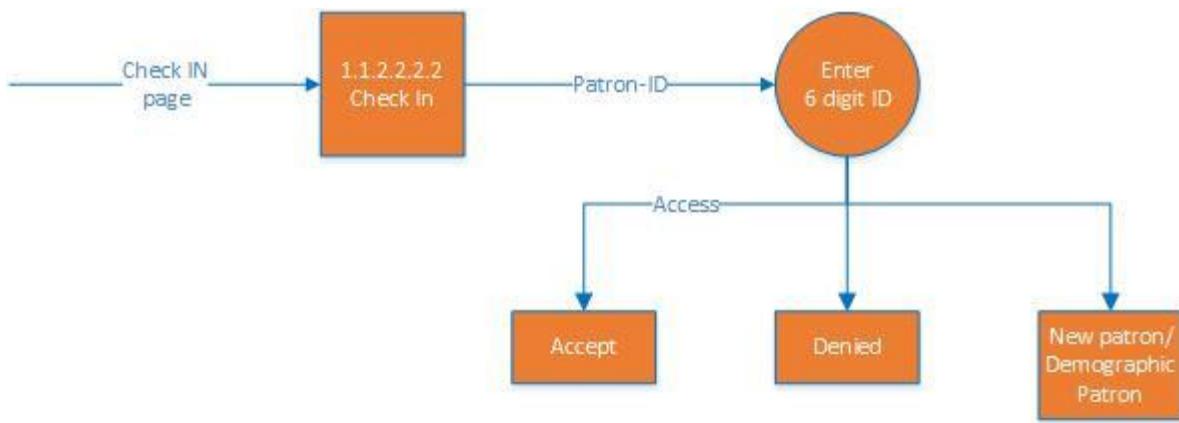
	222	
	Patron ID <6	
	Patron ID=6	
	Patron >6	
<b>Action</b>	Click submit	Check page with details

Subfunction	Input	Message	Output
<b>Username</b>	Appala Chekuri		Show name on the screen
<b>Waiver form</b>	Accept with signature		Submit
<b>Action</b>	Click Submit		Added as the demographic user

#### 4.3.4 Technique 2: Negative Testing

Subfunction	Input	Message	Output
<b>Patron ID</b>	abedcfg	Error	Error window shows information to enter correct 6-digit patron ID
<b>Action</b>	Submit	Error	Will return to Check IN page

#### 4.3.5 State based graph



#### 4.3.6 Merge Test Cases

Test Number	Label	Input	Expected Output	Error message
TC1	Patron ID	734621	Show check IN Details	
	Action	Click submit	Check page with details	
TC2	Username	Appala Chekuri	Show name on the screen	
	Waiver form	Accept with signature	Submit	
	Action	Click Submit	Added as the demographic user	
TC3	Patron ID	abedcfg	Error	Error window shows information to enter correct 6-digit patron ID
	Action	Submit	Error	Will return to Check IN page
TC3	Patron ID	Not provided		Show check IN Details
		1		
		222		

		Patron ID <6		
		Patron ID=6		
		Patron >6		

### 4.3.7 Requirement Traceability Matrix

FWBS Number	Description	Priority (H,M,L)	Test Condition	Test Case_ID	Phase Of testing
<b>FWBS:1.1.2.2.2.2</b>	Using valid information	H	User get Check In	TC1	Unit component
<b>FWBS:1.1.2.2.2.2</b>	Using invalid information	H	User does not get Check IN	TC2	Unit component

### 4.4 Test 4 Waiver form

#### 4.4.1 Sprint & FWBS

FWBS Number: 1.1.1.1

Sprint Number: S4

#### 4.4.2 Functionality and Procedure

<b>FWBS:1.1.1.1</b>	
<b>Function</b>	<b>Should know what we need to do</b>
<b>Patron Name</b>	The patron will enter the patron name on the waiver form.
<b>Patron ID</b>	The patron ID will be entered on the waiver form. 1. The patron ID should be 6-digit.
<b>Accept/Reject</b>	The patron will be accepted or reject the policies of the rock climbing center 1. If accept then the user will update the waiver form into the database

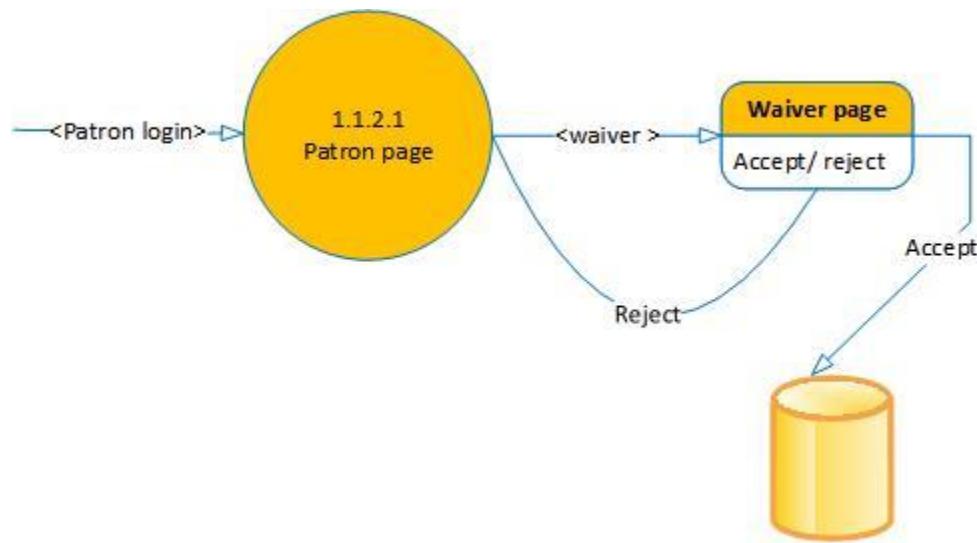
	on his patron ID. 2. If reject, then the user will return to the registration page. So without waiver fill the patron can't enter the rock climbing center.
<b>Signature</b>	The user has to fill the signature with his/her name to submit
<b>Inputs</b>	Patron name, patron ID, accept, Signature
<b>Expected Output</b>	See screenshot for Output

#### 4.4.3 Technique 2: Negative Testing

Subfunction	Input	Message	Output
<b>Patron name</b>	123appalaaaaaaa	Error message	Enter the correct matching patron name
<b>Patron ID</b>	abcd124	Error Message	Enter 6-digit patron ID
<b>Accept</b>	reject	Error Message	Click accept to register successfully
<b>Signature</b>	%773534	Error Message	Enter valid signature
<b>Action</b>	Notsubmit	Error Message	Click submit

#### 4.4.4 Decision Process

#### 4.4.4.1 State Based Graph



#### 4.4.4.2 Merge Test Cases

Test Number	Label	Input	Expected Output	Error Message
TC1	Patron name	Appala Chekuri	See the information on the Screen	
	Patron ID	734621	See the patron ID on the screen	
	Accept	Click Accept	See checkbox Accepted	
	Signature	Appala	Patron name filled	
	Action	Click	Submitted	
TC2	Patron name	123appalaaaaaaaaa	Error message	Enter the correct matching patron name
	Patron ID	abcd124	Error Message	Enter 6-digit patron ID
	Accept	reject	Error Message	Click accept to register successfully
	Signature	%773534	Error Message	Enter valid signature

	Action	Notsubmit	Error Message	Click submit
--	--------	-----------	---------------	--------------

#### 4.4.5 Requirement Traceability Matrix

FWBS Number	Description	Priority (H,M,L)	Test Condition	Test Case_ID	Phase Of testing
<b>FWBS:1.1.1.1</b>	Using valid information	H	User get waiver updated	TC1	Unit component
<b>FWBS:1.1.1.1</b>	Using invalid information	H	User does not get registered	TC2	Unit component

#### 4.5 Test 5. Class Registration

##### 4.5.1 Sprint & FWBS

FWBS Number: 1.1.2.2.1.1

Sprint Number: S5

##### 4.5.2 Functionality and Procedure

<b>FWBS:1.1.2.2.1.1</b>	
<b>Function</b>	<b>What we need to know</b>
<b>Firstname</b>	<p>Enter first name of the patron for Class registration</p> <ol style="list-style-type: none"> <li>If the first name of the user is less than 4 characters, then it displays a message that first name should not be less than 4 characters.</li> <li>It allows successfully the patron to enter the last name if the first name is filled and more than 4 characters.</li> </ol> <p><b>Expected output:</b> See screen Registration 1.</p>
<b>Lastname</b>	<p>Enter last name of the patron for registration.</p> <ol style="list-style-type: none"> <li>If the last name of the user is less than 4 characters, then it displays a message that last name should not be less than 4 characters.</li> <li>It allows successfully the patron to enter the password if the last name is filled and more than 4 characters.</li> </ol> <p><b>Expected output:</b> See screenshot class registration 2</p>

<b>Email</b>	Check the validated email is entered in the field or not 3. If the email is not valid or not according to the format of the email addressing standards, then the registration will not be accepted. 4. The email is entered according to the valid standard then you can get the registration submitted.  <b>Expected output:</b> See the screenshot class registration 6
<b>Phone</b>	The phone number of the patron will be 10-character long 4. If less than 10 characters, then it will show the error that the phone number should be 10-character long. 5. If more than 10 characters, it will show the error that the phone number should be 10 character long. 6. If exactly 10 characters long, then it will be accepted to be submit the registration. <b>Expected output:</b> See screenshot class registration 7.
<b>Semester</b>	Here the patron has to select the semester which he/she want to register for the rock climbing from the dropdown list.
<b>Inputs</b>	See screenshot for input and output of registration.
<b>Expected Output</b>	See screenshot for input and output information

#### 4.5.3 Technique 1: Boundary Value Analysis & Equivalence partitioning

Test case number	First name length	Lastname length	Phone number length
1	0	Not provided	0
2	1	Not provided	1
3	aa	Not provided	123
4	aaa	Not provided	Phone number <10
5	aaaa	Not provided	Phone number =10
6	First name>4	Not provided	Phone number >10
7	0	0	0

8	1	1	1
9	aa	aa	123
10	aaa	aaa	Phone number <10
11	aaaa	aaaa	Phone number =10
12	First name>4	Last name>4	Phone number >10
13	0	0	0
14	1	1	1
15	aa	aa	123
16	aaa	aaa	Phone number <10
17	aaaa	aaaa	Phone number =10
18	First name>4	Last name>4	Phone number >10
19	0	0	0
20	1	1	1
21	aa	aa	123
22	aaa	aaa	Phone number <10
23	aaaa	aaaa	Phone number =10
24	First name>4	Last name>4	Phone number >10
25	0	0	0
26	1	1	1
27	aa	aa	123

28	aaa	aaa	Phone number <10
29	aaaa	aaaa	Phone number =10
30	First name>4	Last name>4	Phone number >10

#### 4.5.4 Technique 2: Negative Testing

Subfunction	Input	Message	Output
<b>Firstname</b>	123444	ERROR message	Enter the correct first name
<b>Lastname</b>	lknd889sd((	ERROR message	Enter the correct last name
<b>Email</b>	@gamil	ERROR message	Enter valid email
<b>Phone</b>	abcdrrfff	ERROR message	Enter valid 10 digit number
<b>Semester</b>	break	ERROR message	Enter valid semester
<b>Action</b>	Not submit	ERROR message	Click submit button for the action to take place

#### 4.5.1.5 Technique 3: Decision Table And Cause-Effective Graph

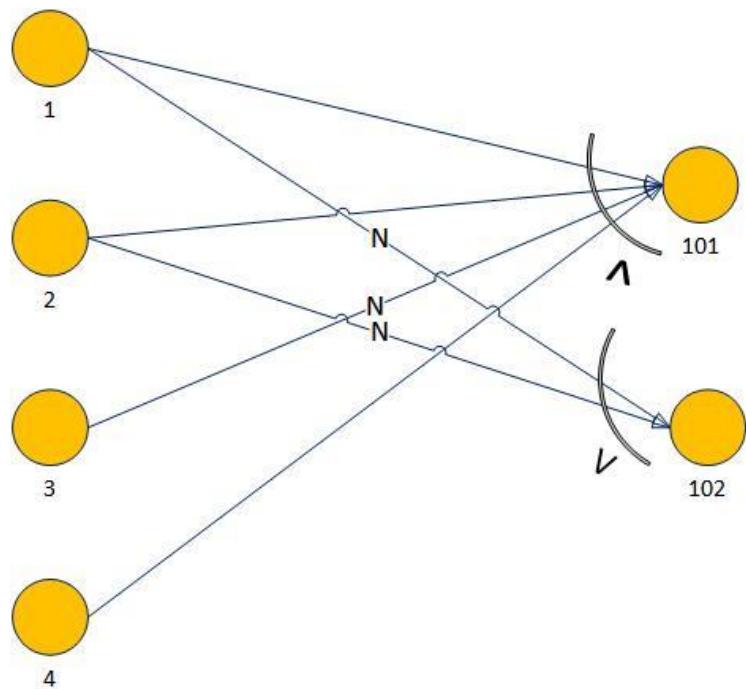
##### 4.5.1.5 .1 Causes:

1. Enter valid information.
2. Enter invalid information
3. Enter wrong data.
4. Click submit to register.

##### 4.5.1.5 .2 Effects:

- 101: return back to registration filing page  
 102: Successfully registered to the class

##### 4.5.1.5 .3 Cause-Effect Graph



#### 4.5.1.5 4 Cause-Effect Table

Process	p1	p2	p3	p4
1	1			
2	0	1		
3			1	
4	1	1	0	1
101	1	0	0	1
102	1	1	1	1

#### 4.5.1.5 5 Merge Test Cases

Test Number	Label	Input	Expected Output	Error message
	Firstname	Appala	See the first name	

TC1	<b>Lastname</b>	Chekuri	See the last name	
	<b>Email</b>	narasimharaju@gmail.com	See the email	
	<b>Phone</b>	3213128444	See phone number	
	<b>Semester</b>	fall	See selected semester	
	<b>Action</b>	Register Class	Class registered	
TC2	<b>Firstname</b>	123444	ERROR message	Enter the correct first name
	<b>Lastname</b>	lknd889sd((	ERROR message	Enter the correct last name
	<b>Email</b>	@gamil	ERROR message	Enter valid email
	<b>Phone</b>	abcdrrfff	ERROR message	Enter valid 10-digit number
	<b>Semester</b>	break	ERROR message	Enter valid semester
	<b>Action</b>	Not submit	ERROR message	Click submit button for the action to take place
TC3	0	Not provided	0	
	1	Not provided	1	
	aa	Not provided	123	
	aaa	Not provided	Phone number <10	
	aaaa	Not provided	Phone number =10	
	First name>4	Not provided	Phone number >10	

	0	0	0	
1	1	1	1	
aa	aa	123		
aaa	aaa	Phone number <10		
aaaa	aaaa	Phone number =10		
First name>4	Last name>4	Phone number >10		
0	0	0	0	
1	1	1	1	
aa	aa	123		
aaa	aaa	Phone number <10		
aaaa	aaaa	Phone number =10		
First name>4	Last name>4	Phone number >10		
0	0	0	0	
1	1	1	1	
aa	aa	123		
aaa	aaa	Phone number <10		
aaaa	aaaa	Phone number =10		
First name>4	Last name>4	Phone number >10		

	0	0	0	
	1	1	1	
	aa	aa	123	
	aaa	aaa	Phone number <10	
	aaaa	aaaa	Phone number =10	
	First name>4	Last name>4	Phone number >10	

#### 4.3.1.5 6.Requirement Traceability Matrix

FWBS Number	Description	Priority (H,M,L))	Test Condition	Test Case_ID	Phase of testing
<b>FWBS:1.1.2.2.2.2</b>	Using valid information	H	User get Check in	TC1	Integration function
<b>FWBS:1.1.2.2.2.2</b>	Using invalid information	H	User does not get Check in	TC2	Integration function

#### 4.6 Test 6. Daily Note

##### 4.6.1 Sprint & FWBS

FWBS Number: 1.1.2.2.2.7

Sprint Number: S6

##### 4.6.2 Functionality and Procedure

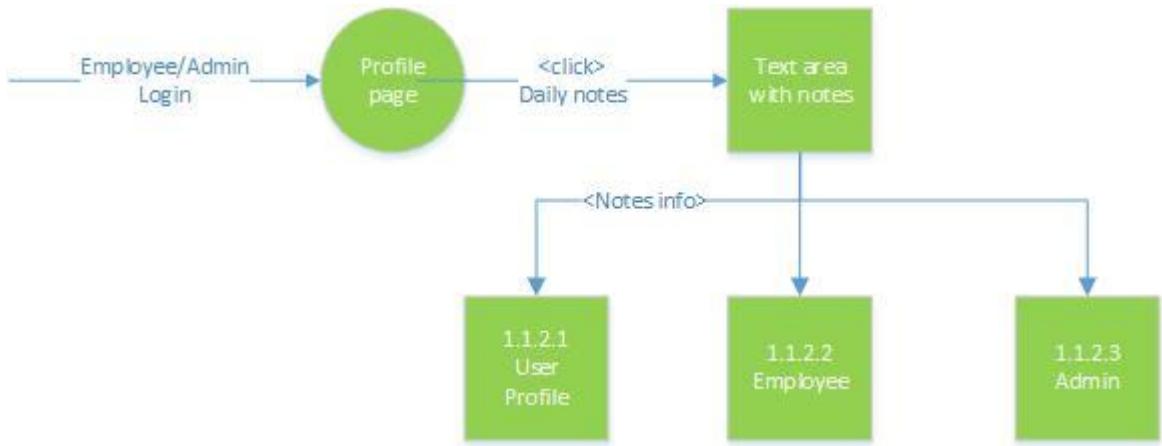
<b>FWBS:1.1.2.2.2.7</b>	
<b>Function</b>	<b>What we need to know</b>

<b>Employee name</b>	The employee has to enter his/her name who want write the daily notes
<b>Daily notes</b>	The daily notes have some text area where the employee will enter some information
<b>To patron</b>	Is the information for the patron then the employee will select the patron checkbox
<b>To admin</b>	Is the information for the admin then the employee will select the admin checkbox
<b>Inputs</b>	Employee name, Daily notes and submit
<b>Expected Output</b>	See screenshot for expected output.

#### 4.6.3 Technique 1: Negative Testing

Subfunction	Input	Message	Output
<b>Employee name</b>	12891234	ERROR message	Enter correct employee name
<b>Daily notes</b>	-----	ERROR message	Enter valid text
<b>To patron</b>	Not selected	ERROR message	Click to send info to patron
<b>To admin</b>	Not selected	ERROR message	Click to send info to admin
<b>Action</b>	Not send	ERROR message	Click send information to take action.

#### 4.6.4 State Based Graph



#### 4.6.5 Merge Test Cases

Test Number	Label	Input	Error message	Expected Output
TC1	<b>Employee name</b>	hussain		See employee name
	<b>Daily notes</b>	Today rock climbing is closed		See notes information
	<b>To patron</b>	Click checkbox		See patron check box checked
	<b>To admin</b>	---		Not checked
	<b>Action</b>	Click send		Notes will send to patron
TC 2	<b>Employee name</b>	12891234	ERROR message	Enter correct employee name
	<b>Daily notes</b>	-----	ERROR message	Enter valid text
	<b>To patron</b>	Not selected	ERROR message	Click to send info to patron
	<b>To admin</b>	Not selected	ERROR message	Click to send info to admin
	<b>Action</b>	Not send	ERROR message	Click send information to take action.

## 4.6.6 Requirement Traceability Matrix

FWBS Number	Description	Priority (H,M,L)	Test Condition	Test Case_ID	Phase Of testing
<b>FWBS:1.1.2.2.2.7</b>	Using valid information	H	Send valid text to patron or admin	TC1	Integration function
<b>FWBS:1.1.2.2.2.7</b>	Using invalid information	H	Not Send valid text to patron or admin	TC2	Integration function
<b>FWBS:1.1.2.2.2.7</b>	Using invalid information	H	Send unvalid text to patron or admin	TC3	Integration function

## 4.7 Test 7. Payment

### 4.7.1 Sprint & FWBS

FWBS Number: 1.1.1.3

Sprint Number: S7

### 4.7.2 Functionality and Procedure

<b>FWBS:1.1.1.3</b>	
<b>Function</b>	<b>What we need to know</b>
<b>Patron ID</b>	The patron ID is used for the updating the payment information. The payment information
<b>Payment information</b>	Time of payment
<b>Employee name</b>	Employee who is present at the time of patron payment
<b>Payment status</b>	Payment done or not done
<b>Inputs</b>	Dropdown box to say the payment done or not
<b>Expected Output</b>	Patron payment information will be updated to the database.

### 4.7.3 causes and effects

#### 4.7.3.1 Causes:

1. Payment done
2. Payment not done
3. Membership has expired.

#### 4.7.3.2 Effects:

- 101: The patron will not be allowed to enter the rock climbing center.  
102: The patron can enter the rock climbing center.

### 4.7.4 Technique 1: Negative Testing

Subfunction	Input	Message	Output
Patron ID	128agags	ERROR message	Enter correct patron ID
Payment information	-----	ERROR message	Enter valid details
Employee name	Not selected	ERROR message	Enter employee name
Payment status	Not selected	ERROR message	Select the correct payment status
Action	Not send	ERROR message	Click update information to take action.

### 4.7.5 Merge Test Cases

Test Number	Label	Input	Expected Output	Error message
TC1	Patron ID	734621	See patron ID	
	Payment information	17th-Dec, 2012 5:15 pm	See payment information	
	Employee name	Hussain	See Employee name	

	<b>Payment status</b>	Done	Selected status	
	<b>Action</b>	Click update	Patron payment updated	
TC2	<b>Patron ID</b>	128agags	ERROR message	Enter correct patron ID
	<b>Payment information</b>	-----	ERROR message	Enter valid details
	<b>Employee name</b>	Not selected	ERROR message	Enter employee name
	<b>Payment status</b>	Not selected	ERROR message	Select the correct payment status
	<b>Action</b>	Not send	ERROR message	Click update information to take action.

## 4.8 Test 8 Schedule class

### 4.8.1 Sprint & FWBS

FWBS Number: 1.1.2.2.1.2.1

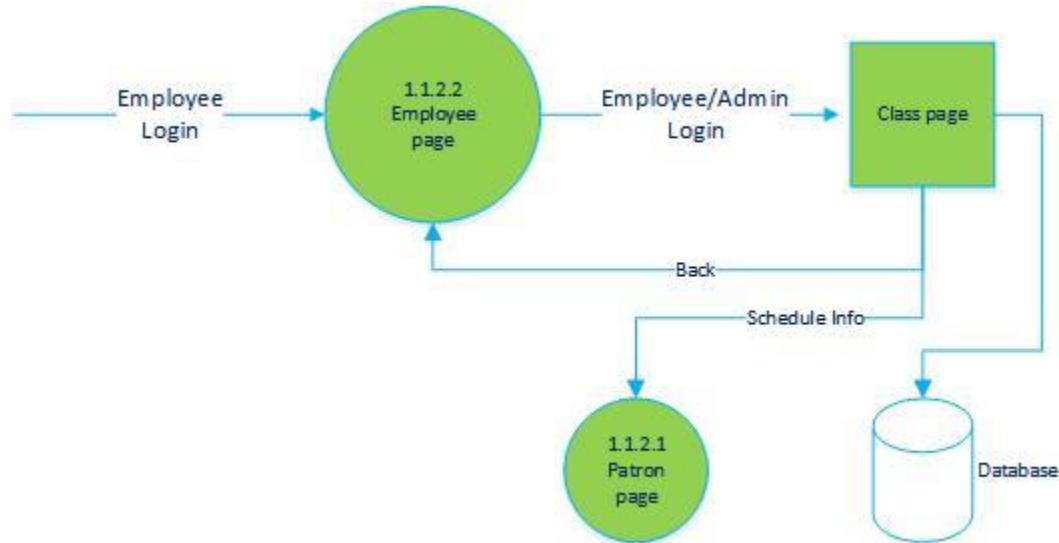
Sprint Number: S8

### 4.8.2 Functionality and Procedure

<b>FWBS:1.1.2.2.1.2.1</b>	
<b>Function Description</b>	<b>Procedure</b>
<b>Employee_name</b>	Enter first name of the patron for Class schedule 3. If the first name of the user is less than 4 characters, then it displays a message that first name should not be less than 4 characters. 4. It allows successfully the patron to enter
<b>class_name</b>	Select class name from list
<b>Schedule time</b>	Select time by scroll down for hour, and select AM or PM
<b>Checked box 1</b>	Click on “patron” box to send to patron

<b>Checked box 2</b>	Click on "admin" box to send to admin
<b>Inputs</b>	Class_name, time and one action either patron or admin
<b>Expected Output</b>	See screenshot

#### 4.8.3 State Based Graph



#### 4.8.4 Technique 1: Boundary Value analysis & Equivalence partitioning

Test case	Employee name length	Classname length	Schedule time Info length
1	Not provided	Not provided	Not provided
2	1	Not provided	Not provided
3	a	Not provided	Not provided
4	aa	Not provided	Not provided
5	aaa	Not provided	Not provided

6	Employee name<4	Not provided	Not provided
7	Employee name>15	Not provided	Not provided
8	Employee name=16	Not provided	Not provided
9	Not provided	0	Not provided
10	1	1	Not provided
11	a	a	Not provided
12	aa	aa	Not provided
13	aaa	aaa	Not provided
14	Employee name<4	Classname <4	Not provided
15	Employee name>15	class name >20	Not provided
16	Employee name=16	Classname = 21	Not provided
17	Not provided	0	0
18	1	1	1
19	a	a	0-12 hours
20	aa	aa	12-24hours
21	aaa	aaa	0
22	Employee name<4	Classname <4	1
23	Employee name>15	class name >20	0-12 hours
24	Employee name=16	Classname = 21	12-24hours

#### 4.8.5 Technique 2: Negative Testing

Subfunction	Input	Message	Output
<b>Employee name</b>	123appalaaaaaaaa	Error message	Enter the correct matching patron name
<b>class_name</b>		Error Message	Please select from list
<b>Schedule time</b>	8	Error Message	Select AM or PM
<b>To patron</b>		Error Message	Unchecked box
<b>To admin</b>		Error Message	unchecked box
<b>Action</b>	Not submit	Error Message	Click submit

#### 4.8.6 Merge Test Cases

Test Number	Label	Input	Expected Output	
TC1	<b>Employee name</b>	Hussain		Hussain
	<b>class_name</b>	Click		See class name wall climbing
	<b>Schedule time</b>	Click		See time 8pm
	<b>To patron</b>	Check		Checked
	<b>To admin</b>	Check		Checked
	<b>Action</b>	Click		Click submit
TC2	<b>Employee name</b>	123appalaaaaaaaa	Error message	Enter the correct matching patron name

TC3	<b>class_name</b>		Error Message	Please select from list
	<b>Schedule time</b>	8	Error Message	Select AM or PM
	<b>To patron</b>		Error Message	Unchecked box
	<b>To admin</b>		Error Message	unchecked box
	<b>Action</b>	Not submit	Error Message	Click submit
	<b>Employee name length</b>	<b>Classname length</b>	<b>Schedule time Info length</b>	
	Not provided	Not provided	Not provided	
	1	Not provided	Not provided	
	a	Not provided	Not provided	
	aa	Not provided	Not provided	
	aaa	Not provided	Not provided	
	Employee name<4	Not provided	Not provided	
	Employee name>15	Not provided	Not provided	
	Employee name=16	Not provided	Not provided	
	Not provided	0	Not provided	
	1	1	Not provided	
	a	a	Not provided	
	aa	aa	Not provided	

	aaa	aaa	Not provided	
	Employee name<4	Classname <4	Not provided	
	Employee name>15	class name >20	Not provided	
	Employee name=16	Classname = 21	Not provided	
	Not provided	0	0	
	1	1	1	
	a	a	0-12 hours	
	aa	aa	12-24hours	
	aaa	aaa	0	
	Employee name<4	Classname <4	1	
	Employee name>15	class name >20	0-12 hours	
	Employee name=16	Classname = 21	12-24hours	

#### 4.8.7 Requirement Traceability Matrix

FWBS Number	Description	Priority (H,M,L)	Test Condition	Test Case_ID	Phase Of testing
1.1.2.2.1.2.1	Using valid information	H	User get register in class	TC1	Integration function
1.1.2.2.1.2.1	Using invalid information	H	User does not get registered in class	TC2	Integration function

## 4.9 Test 9 Send notification

### 4.9.1 Sprint & FWBS

FWBS Number: 1.1.2.2.1.2.2

Sprint Number: S9

### 4.9.2 Functionality and Procedure

FWBS:1.1.2.2.1.2.2	
Function Description	What we need to know
<b>Patron name</b>	List of patron names will be added to the notification information individually.
<b>Patron list</b>	The patron list will be selected if the information need to be send to all patron.
<b>Employee name</b>	The Employee name will be selected from the list of employees who need to know the information.
<b>Employee list</b>	The list of employees will be selected to the notification bar.
<b>Admin</b>	The admin has the full access to the send notification to the people in the rock climbing center.
<b>Notification</b>	The information will be typed and emailed to the destination.
<b>Inputs</b>	Patron name, patron list , employee name ,employee list, Notification information.
<b>Expected Output</b>	See the notification on the destination email.

### 4.9.3 Technique 1: Negative Testing

Subfunction	Input	Message	Output
<b>Patron name</b>	11234	Error message	Enter correct patron name
<b>Patron list</b>	radiobutton	Error message	Click the patron list

<b>Employee name</b>	11234	Error message	Enter correct employee list
<b>Employee list</b>	-----	Error message	Click the employee list
<b>Admin</b>	No name	Error message	Enter admin name
<b>Notification</b>	-----	Error message	Enter some information to send as notification
<b>Action</b>		Error message	Click send to send notification.

#### 4.9.4 Merge Test Cases

<b>Test Number</b>	<b>Label</b>	<b>Input</b>	<b>Expected Output</b>	<b>Error message</b>
TC1	<b>Patron name</b>	Appala	Show the name on the screen	
	<b>Patron list</b>	Click patron list check box	See patron list clicked	
	<b>Employee name</b>	Hussain	Show employee name on the screen	
	<b>Employee list</b>	Click employee list check box	See employee list clicked	
	<b>Admin</b>	Shaohu	See admin name	
	<b>Notification</b>	Hi hello everyone	See the information of notification.	
	<b>Action</b>	Send	Click send	
	<b>Patron name</b>	11234	Error message	Enter correct patron name
	<b>Patron list</b>	radiobutton	Error message	Click the patron list
	<b>Employee name</b>	11234	Error message	Enter correct employee list

TC2	<b>Employee list</b>	-----	Error message	Click the employee list
	<b>Admin</b>	No name	Error message	Enter admin name
	<b>Notification</b>	-----	Error message	Enter some information to send as notification
	<b>Action</b>		Error message	Click send to send notification.

#### 4.9.5 Requirement Traceability Matrix

FWBS Number	Description	Priority (H,M,L))	Test Condition	Test Case_ID	Phase Of testing
<b>FWBS:1.1.2.2.2.1.2.2</b>	Using valid information	H	Enter valid patron list	TC1	Integration function
<b>FWBS:1.1.2.2.2.1.2.2</b>	Using valid information	H	Enter valid employee list	TC2	Integration function
<b>FWBS:1.1.2.2.2.1.2.2</b>	Using invalid information	H	Enter valid admin information	TC3	Integration function
<b>FWBS:1.1.2.2.2.1.2.2</b>	Using invalid information	H	Click action button.	TC4	Integration function

### 4.10 Test 10 Certification management

#### 4.10.1 Sprint & FWBS

FWBS Number: 1.1.2.2.1.3

Sprint Number: S9

#### 4.10.2 Functionality and Procedure

FWBS:1.1.2.2.1.3	
Function	What we need to know
Level of Certification	Describe level of certificate which are level 0 to level 3
Patron_name	The patron will enter the patron name on the certification
PDF_File	The patron has to upload the certificate as PDF file format
Submit	Click action after upload is completed
Inputs	Patron name, PDF_file, submit
Expected Output	See screenshot for Output

#### 4.10.3 Technique 2: Negative Testing

Subfunction	Input	Message	Output
Patron name	123appalaaaaaaaa	Error message	Enter the correct matching patron name
File	appala.txt	Error Message	File format must be PDF format
Action	reject	Error Message	Click accept to register successfully

#### 4.10.4 Merge Test Cases

Test Number	Label	Input	Expected Output
TC1	Patron name	Appala Chekuri	See the information on the Screen
	File	Appala.PDF	File is uploaded successfully
	Action	Click submit	Submit done
TC2	Patron name	123appalaaaaaaaa	Error message
	File	appala.txt	Error Message

	Action	reject	Error Message
--	--------	--------	---------------

#### 4.10.5 Requirement Traceability Matrix

FWBS Number	Description	Priority (H,M,L))	Test Condition	Test Case_ID	Phase Of testing
FWBS:1.1.2.2.1.3	Using valid information	H	User get certification verified	TC1	Integration function
FWBS:1.1.2.2.1.3	Using invalid information	H	User does not get certificate verified	TC2	Integration function

#### 4.11 Test 11 Request Admin for suspension

##### 4.11.1 Sprint & FWBS

FWBS Number: 1.1.2.2.1.2.2

Sprint Number: S11

##### 4.11.2 Functionality and Procedure

FWBS: 1.1.2.2.1.2.2	
Function	What we need to know
Employee_name	Type in employee name
Patron_name	Type in patron name
Patron_id	Type in patron id
Reason of suspension	Type reason of suspension
Inputs	Employee_name, Patron_name and Id and reason
Action	Click
Expected Output	Request has been sent, see screenshot

### 4.11.3 State Based Graph



### 4.11.4 Technique 1: Boundary Value Analysis & Equivalence partitioning

Test no	Employ name length	Patron name length	Patron _ID
1	Not provided	0	Not provided
2	1	1	1
3	a	aa	222
4	aa	aaa	Patron ID <6
5	aaa	aaaa	Patron ID=6
6	Employee name<4	First name>4	Patron >6
7	Employee name>15	0	1
8	Employee name=16	1	222

9	Not provided	aa	Patron ID <6
10	1	aaa	Patron ID=6
11	a	aaaa	Patron >6
12	aa	First name>4	1
13	aaa	0	222
14	Employee name<4	1	Patron ID <6
15	Employee name>15	aa	Patron ID=6
16	Employee name=16	aaa	Patron >6
17	Not provided	aaaa	1
18	1	First name>4	222
19	a	0	Patron ID <6
20	aa	1	Patron ID=6
21	aaa	aa	Patron >6
22	Employee name<4	aaa	1
23	Employee name>15	aaaa	222
24	Employee name=16	First name>4	Patron ID <6

#### 4.11.5 Technique 2: Negative Testing

Subfunction	Input	Message	Output
Employee_name	123appalaaaaaaaa	Error	Enter the correct matching patron

		message	name
<b>Patron_name</b>		Error Message	Patron must enter
<b>Patron_id</b>	@#33		Patron id must be number
<b>Reason</b>		Error Message	Box must be filled
<b>Action</b>	reject	Error Message	Click accept to register successfully

#### 4.11.6 Merge Test Cases

Test Number	Label	Input	Message	Expected Output
TC1	<b>Employee_name</b>	hussain		Request has been sent
	<b>Patron_name</b>	Apallaaa		
	<b>Patron_id</b>	4494949		
	<b>Reason</b>	Break one of rule		
	<b>Action</b>	Click		
TC2	<b>Employee_name</b>	123appalaaaaaaaa	Error message	Enter the correct matching patron name
	<b>Patron_name</b>		Error Message	Patron must enter
	<b>Patron_id</b>	@#33	Error message	Patron id must be number
	<b>Reason</b>		Error Message	Box must be filled
	<b>Action</b>	reject	Error Message	Click accept to register successfully
	Not provided	0	Not provided	

TC 3	1	1	1	
	a	aa	222	
	aa	aaa	Patron ID <6	
	aaa	aaaa	Patron ID=6	
	Employee name<4	First name>4	Patron >6	
	Employee name>15	0	1	
	Employee name=16	1	222	
	Not provided	aa	Patron ID <6	
	1	aaa	Patron ID=6	
	a	aaaa	Patron >6	
	aa	First name>4	1	
	aaa	0	222	
	Employee name<4	1	Patron ID <6	
	Employee name>15	aa	Patron ID=6	
	Employee name=16	aaa	Patron >6	
	Not provided	aaaa	1	
	1	First name>4	222	
	a	0	Patron ID <6	
	aa	1	Patron ID=6	

	aaa	aa	Patron >6	
	Employee name<4	aaa	1	
	Employee name>15	aaaa	222	
	Employee name=16	First name>4	Patron ID <6	

## 4.12 Test 12 Suspend patron

### 4.12.1 Sprint & FWBS

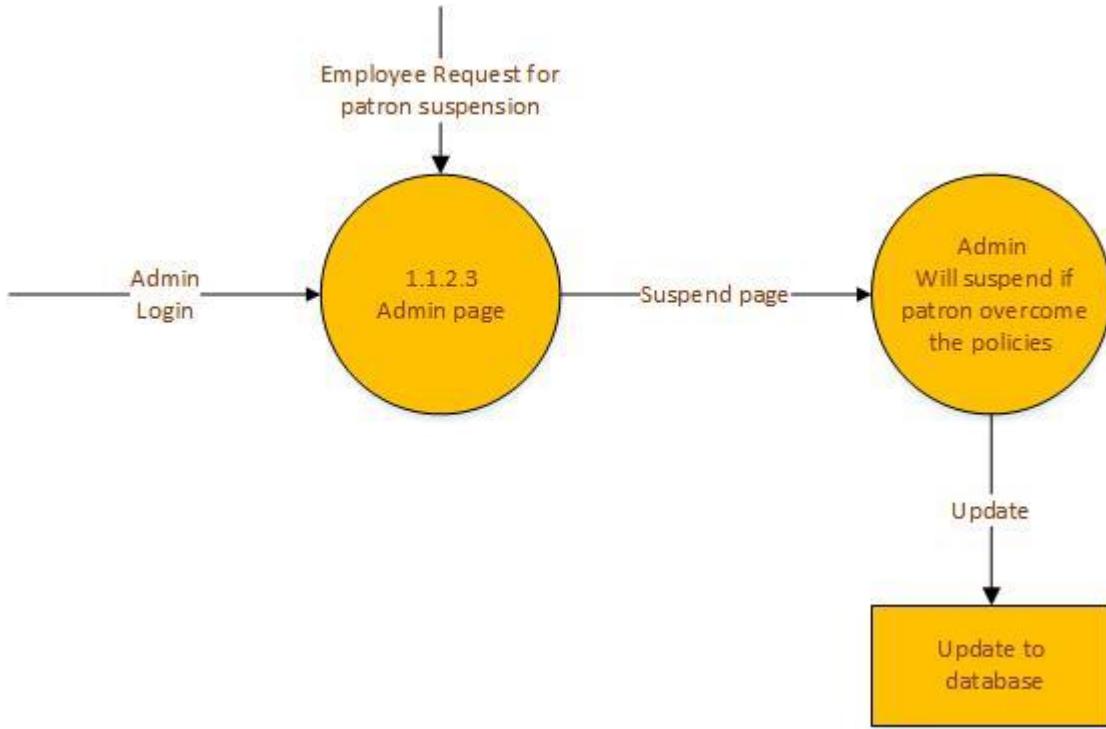
FWBS Number: 1.1.2.3.3

Sprint Number: S12

### 4.12.2 Functionality and Procedure

FWBS:1.1.2.3.3	
Function Description	What we need to know
Employee_name	Type in employee name
Patron_name	Type in patron name
Patron_id	Type in patron id
Period of suspension	Select period of suspension week, month or semester
Action	Click accept
Action	Click reject
Reason of reject	Fill out box if you click reject
Inputs	Employee_name, patron_name and ID, and period of suspension with accept or reject
Expected Output	Two cases will be output, 1- if it reject will display reason of suspension 2- if it accepts, patron will be suspended

### 4.12.3 State Based Graph



### 4.12.4 Negative Testing

Subfunction	Input	Message	Output
<b>Employee_name</b>	123appalaaaaaaaa	Error message	Enter the correct matching patron name
<b>Patron _name</b>		Error Message	Patron must enter
<b>Patron_id</b>	@#33		Patron id must be number
<b>Period of suspension</b>		Error Message	Box must be filled
<b>Action</b>	reject	Error Message	Click accept to register successfully

#### 4.12.5 Merge Test Cases

Test Number	Label	Input	Expected Output	
TC1	<b>Employee_name</b>	Hussain		Hussain has been suspended
	<b>Patron_name</b>	Apalla		
	<b>Patron_id</b>	4494949		
	<b>Period of suspension</b>	week		
	<b>Action</b>	Click accept		
TC2	<b>Employee_name</b>	Hussain		Give first warning
	<b>Patron_name</b>	Apalaaa		
	<b>Patron_id</b>	4494949		
	<b>Period of suspension</b>			
	<b>Action</b>	Click reject		
TC3	<b>Employee_name</b>	123appalaaaaaaaa	Error message	Enter the correct matching patron name
	<b>Patron_name</b>		Error Message	Patron must enter
	<b>Patron_id</b>	@#33		Patron id must be number
	<b>Period of suspension</b>		Error Message	Box must be filled
	<b>Action</b>	reject	Error Message	Click accept to register successfully

## 4.12.6 Requirement Traceability Matrix

FWBS Number	Description	Priority (H, M, L)	Test Condition	Test Case_ID	Phase Of testing
1.1.2.3.3 1.1.2.2.1.2.2	Using valid information	H	Admin get request for suspension	TC1 TC2	Integration
1.1.2.3.3 1.1.2.2.1.2.2	Using invalid information	H	Admin does not get valid data for suspension	TC3	Integration

## 4.13 Test 13. Report generation

### 4.13.1 Sprint & FWBS

FWBS Number: 1.1.2.3.5

Sprint Number: S14

### 4.13.2 Functionality and Procedure

FWBS:1.1.2.3.5	
Function	What we need to know
Report	The report of the patron visiting the rock climbing center
Type of report	There are 3 types of patron visiting rock climbing center 1. Registered user 2. Demographic user 3. Not registered user
Time of report	The time of report will be based on the requirement of the admin 1. From time to To time 2. For particular time
Download option	This option will provide the admin to download the file or just read the file
Inputs	Report type, report time, option to download or view the file.
Expected Output	See the output on the folder saved with data.

### 4.13.3 Technique 1: Positive Testing

Subfunction	Input	Message	Output
New patron name	Appala		Show the patron name on the screen
Demographic patron data	Appala Chekuri		See the information on the screen
Registered patron data	734621		See the patron ID on the screen
From time	14th aug-2016		See the form date on the screen
To time	25th nov-2016		See to date on the screen
Particular date	14th aug - 2016		See particular time on the screen
Particular time	5.30 pm		See the particular time on the screen
Action	Click submit		Data requested

#### 4.13.4 Technique 2: Negative Testing

Subfunction	Input	Message	Output
New patron data	-----	ERROR message	Enter valid parton data
Demographic patron data	132445[[]]	ERROR message	Enter valid patron data
Registered patron data	ldkfls&&&	ERROR message	Enter valid patron data
From time	asddf	ERROR message	Enter valid date
To time	&&&&	ERROR message	Enter valid data
Particular date	aaaa	ERROR message	Enter valid time
Particular time	=====	ERROR message	Enter valid time

Action	-----	ERROR message	Enter valid action
--------	-------	---------------	--------------------

#### 4.13.5 Merge Test Cases

Test Number	Label	Input	Expected Output	Error message
TC1	New patron name	Appala	Show the patron name on the screen	
	Demographic patron data	Appala Chekuri	See the information on the screen	
	Registered patron data	734621	See the patron ID on the screen	
	From time	14th aug-2016	See the form date on the screen	
	To time	25th nov-2016	See to date on the screen	
	Particular date	14th aug - 2016	See particular time on the screen	
	Particular time	5.30 pm	See the particular time on the screen	
	Action	Click submit	Data requested	
TC 2	New patron data	-----	ERROR message	Enter valid parton data
	Demographic patron data	132445[]	ERROR message	Enter valid patron data
	Registered patron data	ldkfsls&&&	ERROR message	Enter valid patron data
	From time	asddf	ERROR message	Enter valid date
	To time	&&&&	ERROR message	Enter valid data
	Particular date	aaaa	ERROR message	Enter valid time

	<b>Particular time</b>	=-----	ERROR message	Enter valid time
	<b>Action</b>	-----	ERROR message	Enter valid action

## 4.13. 6 Requirement Traceability Matrix

FWBS Number	Description	Priority (H,M,L))	Test Condition	Test Case_ID	Phase Of testing
1.1.2.3.5	Using valid information	H	Admin get valid information for backup	TC1	Integration function
1.1.2.3.5	Using invalid information	H	Admin does not get valid data for report generation.	TC2	Integration function

## 4.14 Test 14. Inventory

### 4.14.1 Sprint & FWBS

FWBS Number: 1.2

Sprint Number: S15

### 4.14.2 Functionality and Procedure

FWBS:1.2	
Function	What we need to know
<b>Equipment data</b>	The equipment data has the name of equipment used in the rock climbing center
<b>Equipment information</b>	The equipment information has the <ol style="list-style-type: none"> <li>Start date of equipment</li> <li>End date of equipment</li> <li>Status of equipment</li> <li>Condition of equipment</li> </ol>

<b>Level of climbing</b>	Level of climbing will give information for patron to climb in the rock climbing center.
<b>Inputs</b>	Equipment data, equipment information, start date, end date, status , condition .
<b>Expected Output</b>	See the output screen

#### 4.14.3 Technique 2 Negative Testing

Subfunction	Input	Message	Output
<b>Equipment data</b>	11111	Error message	Enter the correct data
<b>Start date</b>	12t%&*	Error message	Enter the correct date
<b>End date</b>	12t*&%\$#	Error message	Enter the correct date
<b>status</b>	-----	Error message	Enter the correct status
<b>condition</b>	-----	Error message	Enter the valid condition
<b>Level of climbing</b>		Error message	Enter valid value
<b>Action</b>	Click	Error message	Click valid action

#### 4.14.4 Merge Test Cases

Test Number	Label	Input	Expected Output	Error message
TC1	<b>Equipment data</b>	harness	See the equipment name on screen	
	<b>Start date</b>	12th Aug - 2014	See the equipment start date on the screen	
	<b>End date</b>	12th aug- 2017	See the equipment end date on the screen	
	<b>status</b>	working	See the status on the screen	

	<b>condition</b>	good	See the equipment condition on the screen.	
	<b>Level of climbing</b>	Level 1	See the level of climbing of the patron	
	<b>Action</b>	Click submit	Click sumbit	
TC2	<b>Equipment data</b>	11111	Error message	Enter the correct data
	<b>Start date</b>	12t%&*	Error message	Enter the correct date
	<b>End date</b>	12t*&%\$#	Error message	Enter the correct date
	<b>status</b>	-----	Error message	Enter the correct status
	<b>condition</b>	-----	Error message	Enter the valid condition
	<b>Level of climbing</b>		Error message	Enter valid value
	<b>Action</b>	Click	Error message	Click valid action

## 5. NON-FUNCTIONAL REQUIREMENT TESTING

### 5.1 Compatibility test

The following items are tested during the compatibility test for the e-Climbing Software.

- Hardware Compatibility Test
- Operating System Compatibility Test
- Software Compatibility Test
- Database Compatibility Test

Test No.	Test Place	Test Date	Tester
1	Nester center	11/28/2016	Shaohu

Description	System tested on	input	Expected output	Pass/Not Pass	Test case

Hardware Compatibility Test	Lenovo T540p	Software setup	Software should be installed and run correctly		Test 1
Operating System Compatibility Test	Windows 10	Setup the system	Software should be installed and run correctly		Test 1 Test 2
Software Compatibility Test	NetBeans 8.0 Java 7.0	Software setup	Software should be installed and run correctly		Test 5
Database Compatibility Test	MySQL 5.0	Software setup	Data should be stored correctly		Test 6

Test No.	Test Place	Test Date	Tester
2	Nester center	11/28/2016	Hussain

Description	System tested on	input	Expected output	Pass/Not Pass	Test case
Hardware Compatibility Test	Dell Inspiron 5558	Software setup	Software should be installed and run correctly		Test 1
Operating System Compatibility Test	Windows 7	Setup the system	Software should be installed and run correctly		Test 1 Test 2
Software Compatibility Test	NetBeans 8.1 Java 8.1	Software setup	Software should be installed and run correctly		Test 5
Database Compatibility Test	MySQL 5.7	Software setup	Data should be stored correctly		Test 6

## 5.2 Regression test with policy

We will do regression test using the previous merge test cases and automated tools

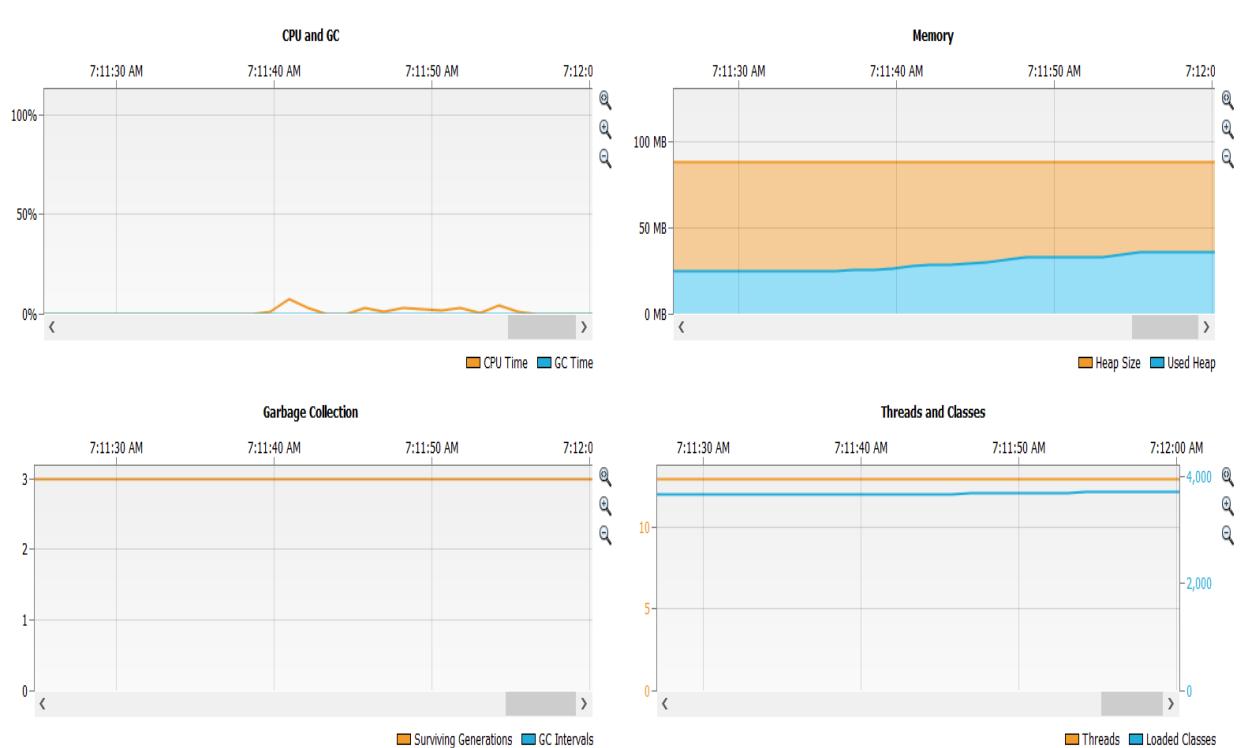
### 5.3 Backup and Recovery test

Test No.	Tests	Input	Expected Results	Pass/Not Pass
1	Shutdown operating system without saving	Restart operating system	The software is restarted correctly	
2	Power Failure	When the system is on, push the power button once	The e-Climbing should be turned on and should work correctly	
3	Hard disks	Server hard disks crashed due to some technical problem	Without data loss	
4	System infected with Malware or virus	Some unexpected results are reflected if user information is extracted unethically	The system will shut down and give warning to users	

### 5.4 Stress and performance test

NetBeans 8.1 or later supports the profile testing tool to replay load testing scripts. Profile reports the software performance in terms of average response time, errors, CPU, memory and thread.

Input	Expected Results
Keep fast input value	Maximum CPU Maximum memory
iterations	CPU consumption memory
Limited RPS(Request per second)	Response time CPU consumption memory
Duration	Response time CPU consumption memory



## 6. UNIT TESTING

Unit testing is a software verification method in which a programmer tests if individual units of source code are fit for use. A unit is the smallest part of the program which may just include a function or procedure.

### 6.1 Test 1 log in

#### Objective:

Test Schedule:

<i>WBS #</i>	<i>Tester</i>	<i>Test Location</i>	<i>Test Date</i>	<i>Developer</i>
1.1.2	Appala	Nester center	11/28/2016	Appala

Pseudocode :

```

1  if(length.username)<6)
2  {
3      system.out.println("error1")
4  }
5 else if(length.username>6)
6  {
7      system.out.println("error2")
8 else
9  {
10     .
11     verify(username)
12     verify(password)
13     if(username &&password)
14     {
15         system.out.println("weclom to home page ")
16     }
17     else
18     {
19         system.out.println("error3")
20     }

```

Coverage Measurement	
Statement Coverage	<b>100%</b>
Branch Coverage	<b>100%</b>
Principal path	<b>100%</b>
Principle Test	<b>70%</b>
Basis Test	<b>100%</b>

### 6.1.1 Statement Coverage

Test No	Username	password	Result
1	H	1234	Username must be six letter
2	#@33hd	T	Username has to be alphanumeric no symbol is accepted
3	@\$Huss	!`3\$	Username has to be alphanumeric no symbol is accepted ,Password must be digit
4	Huss1233	@##@!	Password must be digit
5	#@33hd	1234	Username has to be alphanumeric no symbol is accepted
6	!@mhd	!~#@12	Username has to be alphanumeric no symbol is accepted , Password must be digit

Statement Coverage = Total Statement exercised/ total # of executable statements in program)\*100

$$= 6/6 *100= 100\%$$

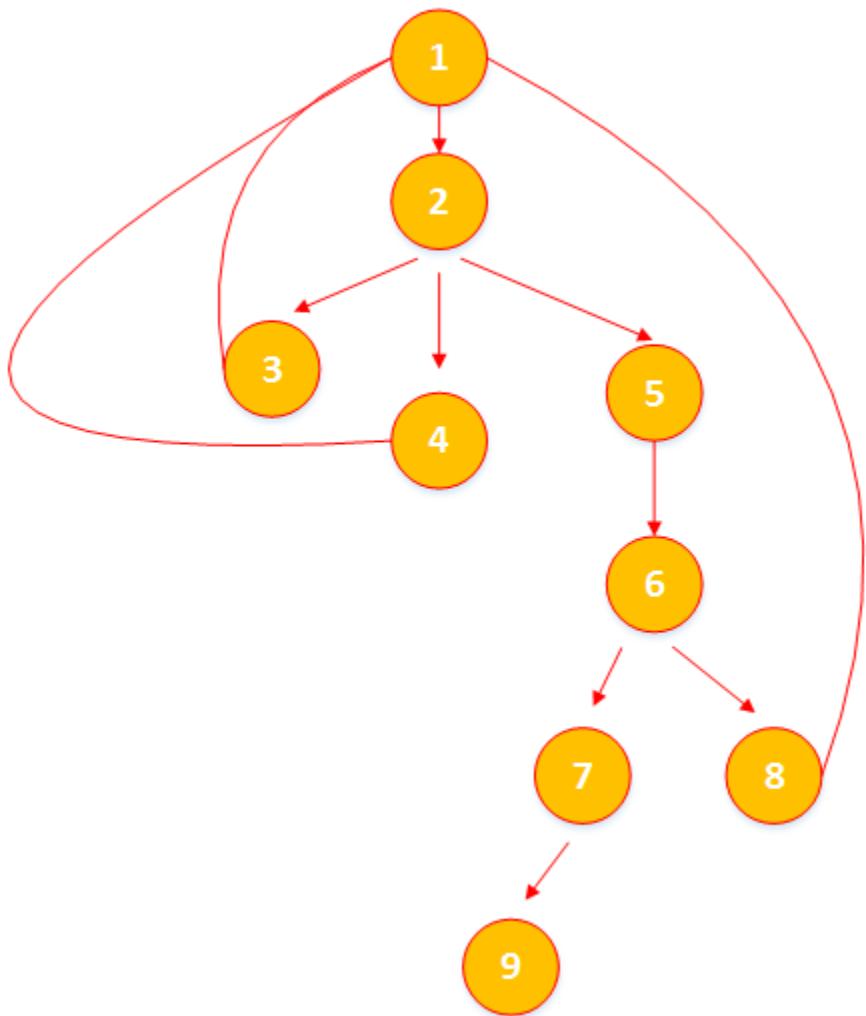
### 6.1.2 Branch Coverage

Inputs	Test case 1 ( all inputs are true)
username	shaohu
password	z1234ss

Inputs	Test case 2 ( all inputs are false)
Username	@shaohu
password	#z1234ss

$$\begin{aligned}\text{Branch Coverage} &= \text{Total decisions exercised / total # of decisions in program)} * 100 \\ &= (2/2) * 100 = 100\%\end{aligned}$$

### 6.1.3 McCabe's



### 6.1.4 Principal path

$$V(G) = 11 - 9 + 2 = 4$$

$$V(G) = 3 + 1 = 4$$

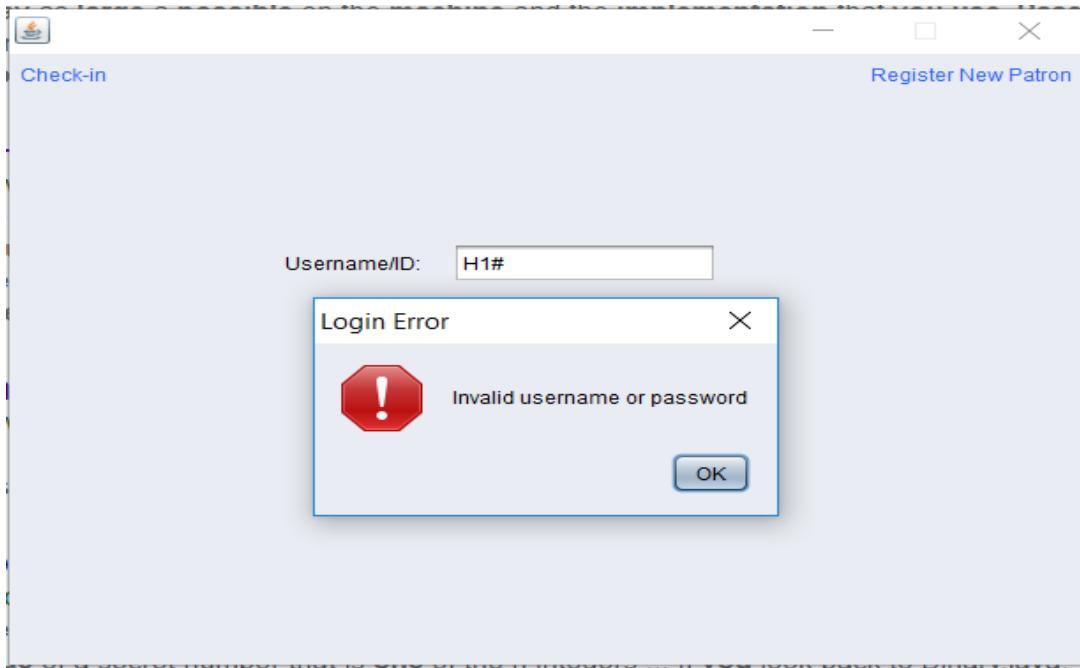
Path #1 : 1-2-3

Path #2: 1-2-4

Path #3: 1-2-5-6-7-9

Path #4: 1-2-5-6-8

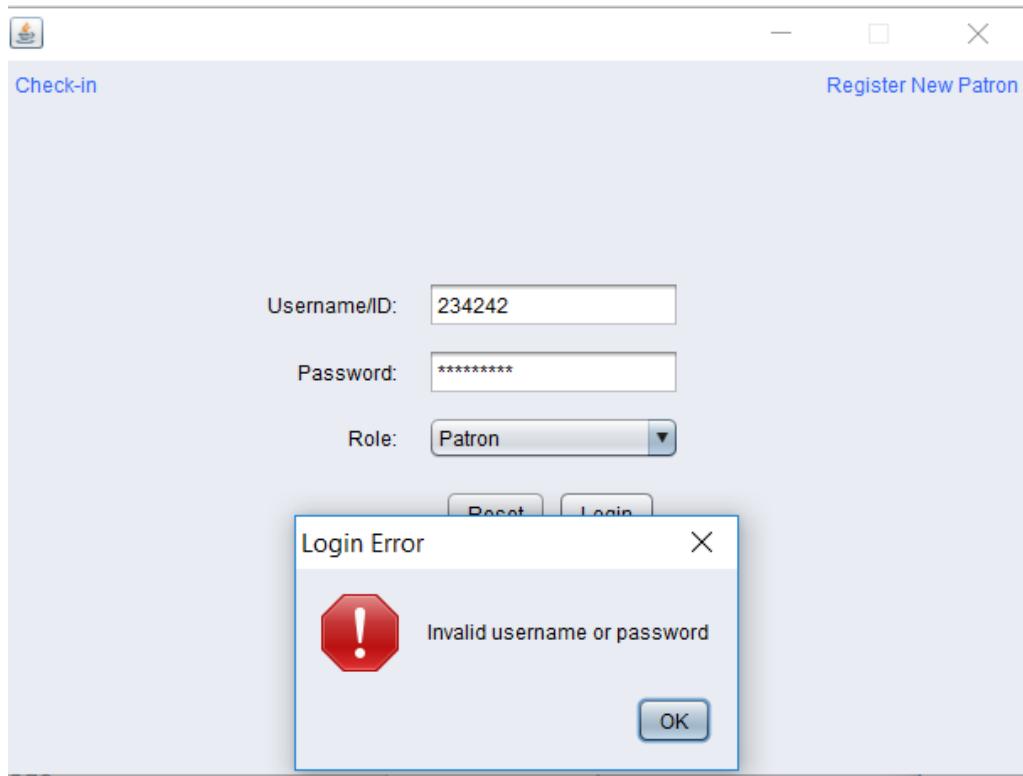
Path #1 : 1-2-3 (username is wrong input)



Screenshot 6.1

Test No	Username	Result
1	H1#	Error:invalid username or password (See Screenshot 6.1)
2	edgdg	Error:invalid username or password
3	((	Error:invalid username or password
4	!!!!	Error:invalid username or password
5	#@5455454546w53	Error:invalid username or password

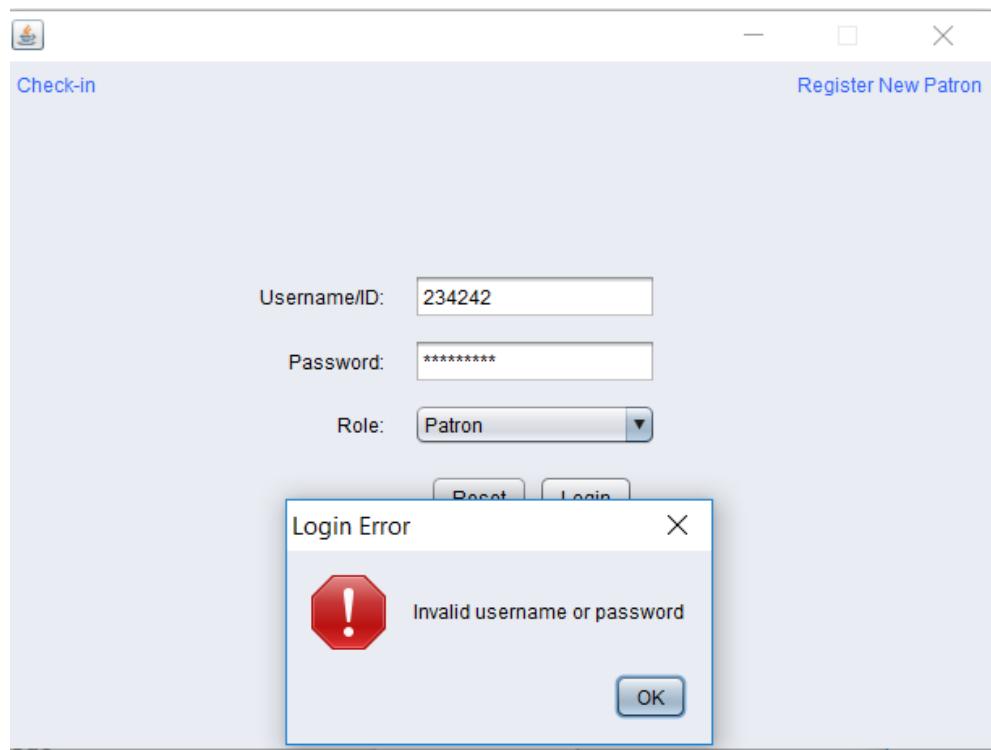
Path #2 :1-2-4 (password is wrong input)



Screenshot 6.2

Test No	Password	Result
1	234242	Error:invalid username or password(see Screenshort 6.2)
2	111!!	Error:invalid username or password
3	?111	Error:invalid username or password
4	!!4!!	Error:invalid username or password
5	5454546w53	Error:invalid username or password

Path #3 :1-2-5-6-7-9(username is true,password is false)



Screenshot 6.3

Test No	User name	Password	Result
1	appala	r3	Error:invalid username or password
2	appala	1r11!!	Error:invalid username or password
3	appala	!!!r!!!!!!!!!!!!!!111	Error:invalid username or password
4	appala	!!r!!	Error:invalid username or password
5	appala	5454546w53	Error:invalid username or password

Path #4 1-2-5-6-8(both true)

Screenshot 6.4

Test No	User name	Password	Result
1	appala	2334	Log in successfully(See Screenshot 6.4)
2	shaohuz	11134	Log in successfully
3	Hussain	34545	Log in successfully
4	sdfsfsdf	354545	Log in successfully
5	zhangs	545454	Log in successfully

## 6.2 Test 2 Registration validation

Objective: The inputs should be validated in correct format.

Test Schedule:

<i>WBS #</i>	<i>Tester</i>	<i>Test Location</i>	<i>Test Date</i>	<i>Developer</i>
1.1.1	Appala	Nester center	11/28/2016	Appala

Pseudocode:

```

1  if (!valid(first_name))
2      print error1;
3  else if(!valid(last_name))
4      print error2;
5  else if(!valid(email))
6      print error3;
7  else if(!valid(password))
8      print error4;
9  else if(conform_pw!=password)
10     print error5;
11 else if(!valid(phone))
12     print error6;
13 else if(!valid(sex))
14     print error7;
15 else if(!valid(DOB))
16     print error8;
17 else
18     print successful;
19

```

Coverage Measurement	
Statement Coverage	<b>50%</b>
Branch Coverage	<b>100%</b>
Principal path	<b>100%</b>
Principle Test	<b>100%</b>
Basis Test	<b>100%</b>

### 6.2.1 Statement Coverage

Test No	first_name	last_name	pass word	con_pw	email	phone	sex	DOB	Result
1	Huss 12	@!ha	@#\$	@#\$	otudi#@g	890,9 87-	-	99- ``12- 16	Last_name must be alphanumeric, password, the email is not correct ,error phone must be 10 digit ,error invalid format for DOB
2	@hay	Otudi	@12	12	h@gmail	605&6	-	19891	first_name must

	y					90&		202	be alphanumeric, password, the email is not correct ,error phone must be 10 digit ,must select gender,error invalid format for DOB
3	@3hus	ot%	123	654	un^@gmail.com	123`1 25	-	12998 8	First name and last name must be alphanumeric, password doesn't matching, the email is not correct ,error phone must be 10 digit ,must select gender,error invalid format for DOB
4	@13	\$FDS	!22	123	#@2@gmai.com	-	0000000 00	20201 212	First name and last name must be alphanumeric, password doesn't matching, the email is not correct ,error phone must be 10 digit ,must select gender,error invalid format for DOB
5	@hus	raju!	F###	F## #	Otudi@gm ail.com	-	444/333/ 333	23/12/ 1988	First name and last name must be alphanumeric, password must be digit. ,error

										phone must be 10 digit ,must select gender,error invalid format for DOB
6	@hus	raju!	- ====9	876 6-	-	male	234,564, 999,88	1989/1 2/04		First name and last name must be alphanumeric, password must be digit, email must be filled ,error phone must be 10 digit ,must select gender,error invalid format for DOB
7	=hyd	%ot#	*&*	u&u	gmail@co me	-	605-690- 8430	12/12/ 1988		First name and last name must be alphanumeric, password must be digit, email must be filled ,error mu st select gender,error invalid format for DOB
8	=hyd	%ot#	*&*	u&u	gmail@co me	-	876/678/ 987	14-04- 1989		First name and last name must be alphanumeric, password must be digit, email must be filled ,error mu st select gender.
9	hussa in	Otudi	1234	123 4	otudi@gma il.com	male	605-690- 8430	18-04- 1989		See screensht with Patron_id
10	hussa	Otudi	\$123	\$12	huss#@gm	-	000/000/	12/04/		Password must

	in			3	ail.com		000	189	be digit, the email is not correct ,error phone must be 10 digit ,error invalid format for DOB
11	Raju	chak ori	1235	123 8	raju@gmai l.com	-	000/000/ 000	12/04/ 189	Password doesn't match, the email is not correct ,error phone must be 10 digit ,error invalid format for DOB
12	Raju	chak ori	1235	123 5	raju@gmai l.com	-	000/000/ 000	12/04/ 189	the email is not correct ,error phone must be 10 digit ,error invalid format for DOB
13	hussa in	Otudi	234	234	raju@gmail .com	-	/000/000	18041 989	Gender must be selected, Phone number is not correct,DOB is not correct format
14	hussa in	Otudi	234	234	raju@gmail .com	male	9889756 3@	18041 989	Phone number is not correct,DOB is not correct format
15	hussa in	Otudi	234	234	raju@gmail .com	male		18041 989	Enter phone number
16	hussa in	ali#	\$#4d	\$#4 d	ra!u@gmail .com	-	9889756 3@	18041 989	Last name must be alphanumeric,gender must be selected,Phone number is not correct,DOB is not correct

										format
17	F%djf g	ali	1234	12#	ra!u@gmail.com	-	98897563@	18041989		Fisrt_name is not correct, password doesn't match , email is not correct,Gender must be selected, Phone number is not correct,DOB is not correct format
18	!skdk s	hadi	1234	1234	ra!u@gmail.com	-	98897563@	18041989		Fisrt_name is not correct, password doesn't match , email is not correct,Gender must be selected, Phone number is not correct,DOB is not correct format
19	_=jdsj	soai	1234	1234	raju@gmail.com	-	605-690-8	18041989		Fisrt_name is not correct, password doesn't match , email is not correct,Gender must be selected, Phone number is not correct,DOB is not correct format
20	dkfkd %	ali	1234	1234	raju@gmail.com	male	605-690-8430@	18041989		Fisrt_name is not correct, password doesn't match , email is not correct,Gender must be selected, Phone number is not

										correct,DOB is not correct format
21	%dfd	shah o	1234	123 4	raju@gmail.com	male	605-690-8430	18041 989		Fisrt_name is not correct, password doesn't match , email is not correct,Gender must be selected,,DOB is not correct format
22	GHF G@	chak ori	1234	123 4	raju@gmail.com	male	605-690-8430	18-04-1989		See screenshot
23	DFdf d!	FD#	1234	123 4	H#@gmail.com	-	/000/012 3	18041 989		last_name is not correct, , email is not correct,Gender must be selected, Phone number is not correct,DOB is not correct format
24	%dfd	dfdf\$	1234	123 4	raju@gmail.com	-	/000/000	18041 989		Fisrt_name and last name is not correct, password doesn't match , email is not correct,Gender must be selected, Phone number is not correct,DOB is not correct format
25	#rrere	^dmf d	1234	123 4	raju@gmail.com	male	9453945 9*	18041 989		Fisrt_name and second name are not correct,, Phone number is not correct,DOB is not correct

										format
26	&kdrf m	&gfdf	1234	123 4	raju@gmail .com	male	605-690- 8430	18041 989		First and Last name must be alphanumeric,n o symbol is not accepted, error in DOB format
27	#fgk	\$feddf	1234 @	123 4	raju@gmail .com	-	605-690- 8430	18041 989		First and Last name must be alphanumeric,n o symbol is not acceptedPassw ord mustbe digit., error in DOB format
28	!233	!otudi	1234 !	123 4	raju@gmail .com	male	/000/000	18041 989		First and Last name must be alphanumeric,n o symbol is not accepted, password must be digit
29	!Fase l	!otudi	@11 2	112	otudi@gma il.com	male	605-690- 8430	18041 989		First and Last name must be alphanumeric,n o symbol is not accepted, password must be digit, error in DOB formate
30	!Fase l	!otudi	@11 2	123	otudi@gma il.com	male	605-690- 8430	18-04- 1989		First and Last name must be alphanumeric,n o symbol is not accepted, password must be digit
31	Huss ain	!otudi	1234	123 4	otudi@gma il.com	male	605-690- 8430	18-04- 1989		Last name must be alphanumeric,n o symbol is not accepted
32	Huss ain	!otudi	@# @w	@# @w	otudi@gma il.com	Male	605-690- 8430	18-04- 1989		Last name must be

									alphanumeric, no symbol is not accepted , password must be digit
--	--	--	--	--	--	--	--	--	--

Statement Coverage = Total Statement exercised/ total # of executable statements in program)\*100  
= 32/64 \*100= 50%

### 6.2.2 Branch Coverage

Inputs	Test case 1 ( all inputs are true)
first_name	appala
last_name	chekuri
password	chap123
conform_password	chap123
phone	32131284311
Semester	Fall
sex	Male
DOB	1992/10/12
Output	See screenshot 11

The screenshot shows a Java Swing application window titled "User Registration". The window contains the following fields:

- First Name: appala
- Last Name: chekuri
- User Name: hussainra
- Password: \*\*\*\*
- Confirm Password: \*\*\*\*
- Email: ala.chekuri@sdstate.edu
- Phone: 32131284311
- Semester: Fall
- Sex: Male
- Date Of Birth: 1992/10/12

At the bottom of the window, there are "Reset" and "Submit" buttons.

screenshot 11

Inputs	Test case 2 ( all inputs are true)
first_name	ap
last_name	888
password	husssssssss
conform_password	z1234ssss
phone	absdrrr45
sex	male
DOB	nafighe
Output	See screenshot

<< Back

First Name: ap

Last Name: 888

User Name: hu

Password: [REDACTED]

Confirm Password: [REDACTED]

Email: appala.cstate.edu

Phone: abcdrrr45

Semester: Fall

Sex: Male

Date Of Birth: nafighe

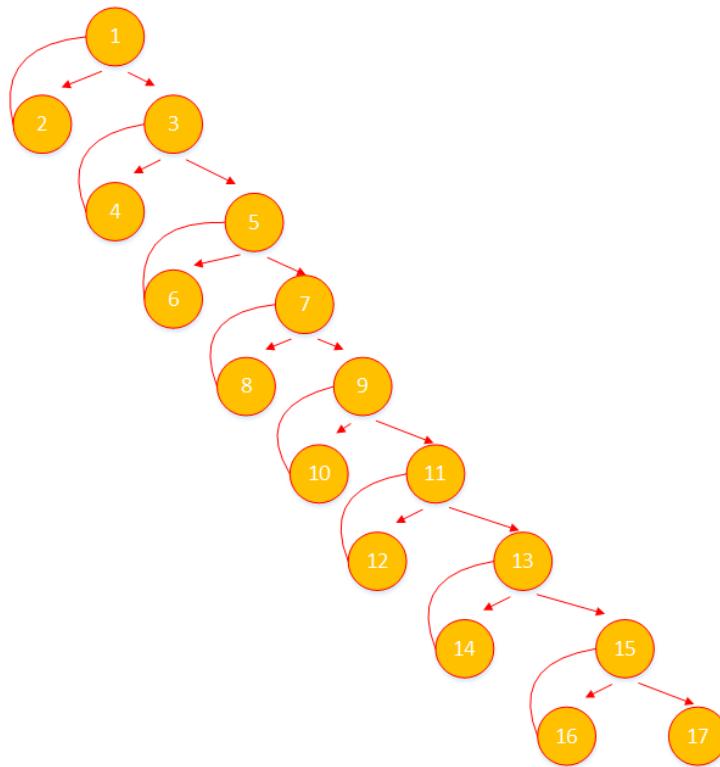
Format (YYYY-MM-DD)

Reset Submit

Screenshot 12

Branch Coverage = Total decisions exercised / total # of decisions in program)\*100  
 $= (2/2)*100 = 100\%$

### 6.2.3 McCabe's



### 6.2.4 Principal path

$$V(G) = 24 - 17 + 2 = 9$$

$$V(G) = 8 + 1 = 9$$

Path #1: 1-2

Path #2: 1-3-4

Path #3: 1-3-5-6

Path #4: 1-3-5-7-8

Path #5: 1-3-5-7-9-10

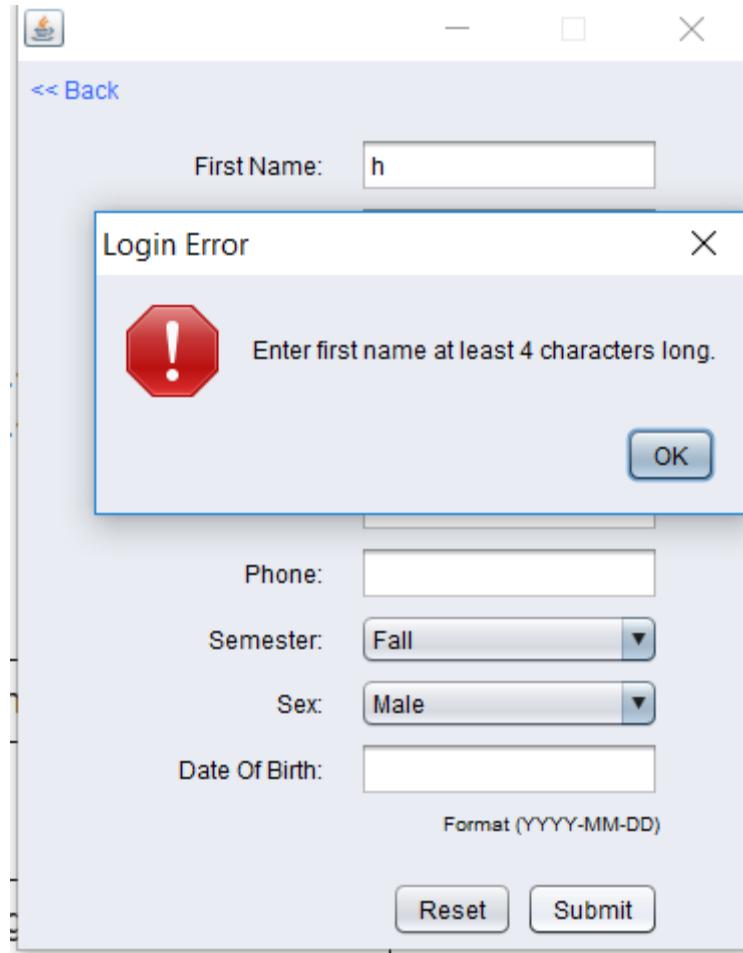
Path #6: 1-3-5-7-9-11-12

Path #7: 1-3-5-7-9-11-13-14

Path #8: 1-3-5-7-9-11-13-15-16

Path #9: 1-3-5-7-9-11-13-15-17

Path #1: 1-2



Screenshot 6.1

Test No	first_name	Result
1	h	Warning:first_name at least 4 character long(see Screenshot 6.1)
2	e	Warning:first_name at least 4 character long
3	54RTET	Warning:first_name has to be alphanumeric no symbol is accepted
4	!!UY!!	Warning:first_name has to be alphanumeric no symbol is accepted
5	#@54w53	Warning:first_name has to be alphanumeric no symbol is accepted

Path #2:1-3-4



Screenshot 6.2

Test No	First Name	Last Name	Result
1	shaohu	z	Warning:enter last name at least 4 characters long (See Screenshot 6.2)
2	li	sdg	Warning:enter last name at least 4 characters long
3	hell	54RT86T	Warning:last_name has to be alphanumeric no symbol is accepted
4	kitty	!9889!	Warning:last_name has to be alphanumeric no symbol is accepted
5	wang	#zhang	Warning:last_name has to be alphanumeric no symbol is accepted

Path #3:1-3-5-6

The screenshot shows a user registration form with fields for First Name, Last Name, User Name, Sex, and Date Of Birth. A modal dialog box titled "Login Error" is displayed, indicating that the user name must be at least 8 characters long. The "OK" button is visible in the dialog.

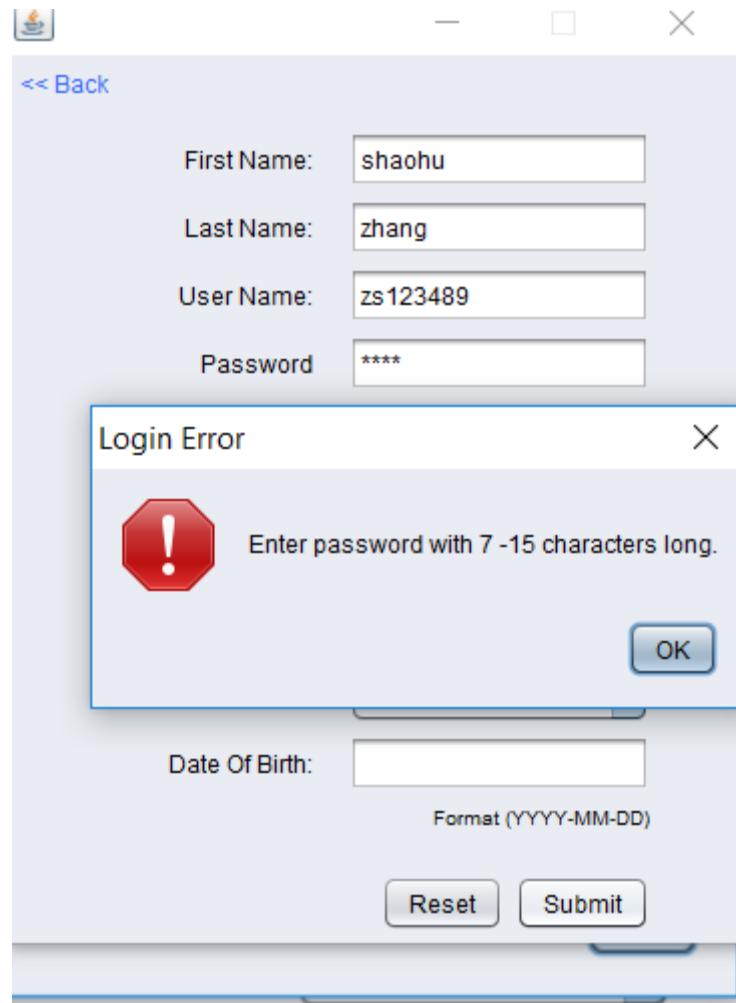
First Name:	shaohu
Last Name:	zhang
User Name:	zs123
Sex:	Male
Date Of Birth:	

Format (YYYY-MM-DD)

Screenshot 6.3

Test No	First Name	Last Name	User name	Result
1	Shaohu	Zhang	zs123	Warning:enter user name at least 8 characters long (See Screenshot 6.3)
2	Shaoh	Zang	&zhang23	Warning:user name has to be alphanumeric no symbol is accepted
3	Shaoh	Zhang	zhang!3?	Warning:user name has to be alphanumeric no symbol is accepted
4	Saohu	Zang	*zhang23	Warning:user name has to be alphanumeric no symbol is accepted
5	Shhu	hang	zg23?%%	Warning:last_name has to be alphanumeric no symbol is accepted

Path #4:1-3-5-7-8



Screenshot 6.4

Test No	First Name	Last Name	User name	Password	Result
1	Shaohu	Zhang	zs123489	1234	Error:enter password with 7-15 characters long(See Screenshot 6.4)
2	Shaoh	Zang	1zhang23456	45	Error:enter password with 7-15 characters long alphanumeric no symbol is accepted
3	Shaoh	Zhang	zhang3	4789	Error:enter password with 7-15 characters long
4	Saohu	Zang	2zhang23	85tf	Error:enter password with 7-15 characters long
5	Shhu	hang	zg2y	re565	Error:enter password with 7-15 characters long

Path #5:1-3-5-7-9-10

The screenshot shows a user registration form. At the top, there are fields for First Name (shaohu), Last Name (zhang), and User Name (zs123489). Below this is a modal dialog titled "Login Error" with a red exclamation mark icon. The message inside says "password and confirm password do not match." There is an "OK" button at the bottom right of the dialog. Below the dialog, there are fields for Sex (Male) and Date Of Birth (with a placeholder "Format (YYYY-MM-DD)"). At the bottom are "Reset" and "Submit" buttons.

Screenshot 6.5

Test No	First Name	Last Name	User name	Password	con_pwd	Result
1	Shaohu	Zhang	zhang23rrr	1234567	123456	Error:password and confirm password don't match

Path #6:1-3-5-7-9-11-12

The screenshot shows a Windows application window titled "Login Error". Inside the window, there is a red octagonal icon with a white exclamation mark. To its right, the text "Enter valid email address." is displayed. At the bottom right of the window is a blue "OK" button. The background of the window is light blue. Below the "Login Error" window, there are two buttons: "Reset" and "Submit".

<< Back

First Name: shaohu

Last Name: zhang

User Name: zs123489

Password: \*\*\*\*\*

Confirm Password: \*\*\*\*\*

Email: shaohu

Login Error

Enter valid email address.

OK

Reset Submit

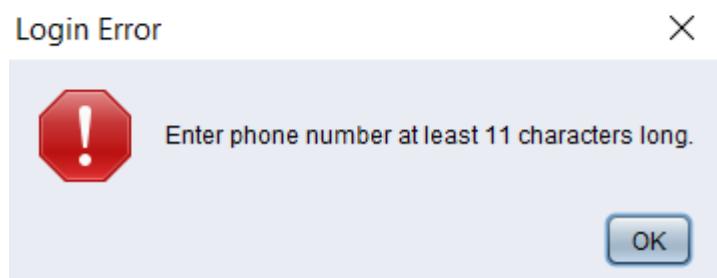
Screenshot 6.6

Test No	First Name	Last Name	User name	Password	con_pw	email	Result
1	shaohu	zhang	zs123489	1234567	1234567	shaohu	Error:Enter valid email address

Path #7:1-3-5-7-9-11-13-14

[<< Back](#)

First Name:	shaohu
Last Name:	zhang
User Name:	zs123489
Password:	*****
Confirm Password:	*****
Email:	shaohu.zhang@sdstate.edu
Phone:	545



Test No	First Name	Last Name	User name	Passwo rd	con_p w	email	phon e	Result
1	shaohu	zhang	zs123489	1234567	1234567	shaohu.zhang@sdstate.edu	545	Warning: user name has to be alphanumeric no symbol is accepted

Path #8:1-3-5-7-9-11-13-15-16

[!\[\]\(b548fc861a69d56ff7d8ca667a96259c\_img.jpg\) << Back](#)

First Name:	shaohu
Last Name:	zhang
User Name:	zs123489

**Login Error** X



Enter phone number at least 11 characters long.

**OK**

Sex:	Male
Date Of Birth:	1987
Format (YYYY-MM-DD)	
<b>Reset</b> <b>Submit</b>	

Test No	User name	Password	con_pw	email	phone	birth	Result
1	zs123489	1234567	1234567	shaohu.zhang@sdstate.edu	54564564	1987	Warning: Enter phone number at least 11 characters long

### 6.3 Test 3 Check-in validation

#### Objective:

To make sure the input value for check in is valid.

Test Schedule:

<i>WBS #</i>	<i>Tester</i>	<i>Test Location</i>	<i>Test Date</i>	<i>Developer</i>
1.1.2.2.2.2	Appala	Nester center	11/28/2016	Appala

Coverage Measurement	
Statement Coverage	<b>100%</b>
Branch Coverage	<b>100%</b>
Principal path	<b>100%</b>
Principle Test	<b>70%</b>
Basis Test	<b>100%</b>

#### 6.3.1 Statement Coverage

Test No	patron_id	new_user	Demographic user	Result
1	123456	0	0	1
2	1234\$2	1	0	1
3	@3444	0	1	1
4	!@43d	0	0	0

Statement Coverage = Total Statement exercised/ total # of executable statements in program)\*100  
 $= (4/4)*100= 100\%$

#### 6.3.2 Branch Coverage

Inputs	Test case 1 ( all inputs are true)
Patron_id	12345
New_user	-
demographic_user	-warning:

Inputs	Test case 2 ( all inputs are true)
Patron_id	-
New_user	click
demographic_user	-

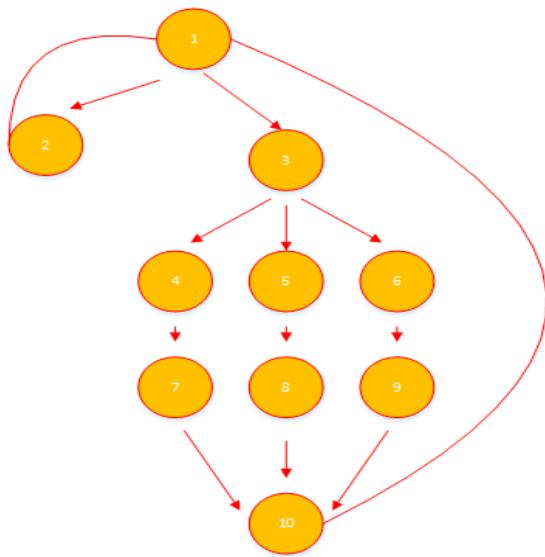
Inputs	Test case 3 ( all inputs are true)
Patron_id	-
New_user	-
demographic_user	click

Inputs	Test case 4 ( all inputs are false)
Patron_id	@~34
New_user	-
demographic_user	-

Inputs	Test case 5 ( all inputs are false)
Patron_id	-
New_user	-
demographic_user	-

$$\begin{aligned}
 \text{Branch Coverage} &= \text{Total decisions exercised / total # of decisions in program)} * 100 \\
 &= (5/5) * 100 = 100\%
 \end{aligned}$$

### 6.3.3 McCabe's



### 6.3.4 Principal path

$$V(G) = 13 - 10 + 2 = 5$$

$$V(G) = 4 + 1 = 5$$

path#1:1-2

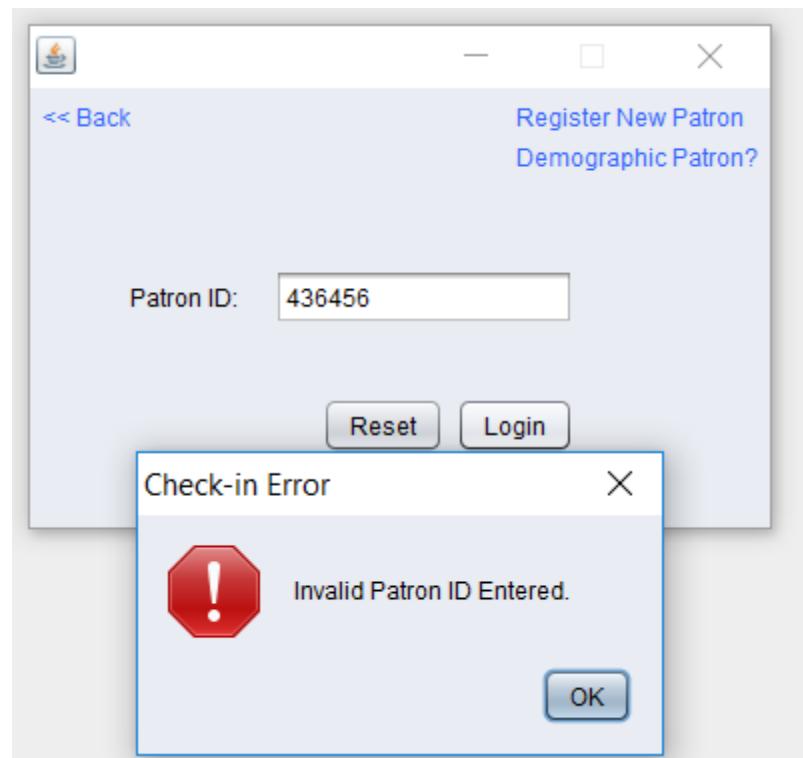
path#2:1-3-4-7-10

path#3:1-3-5-8-10

path#4:1-3-6-9-10

path#5:1-10

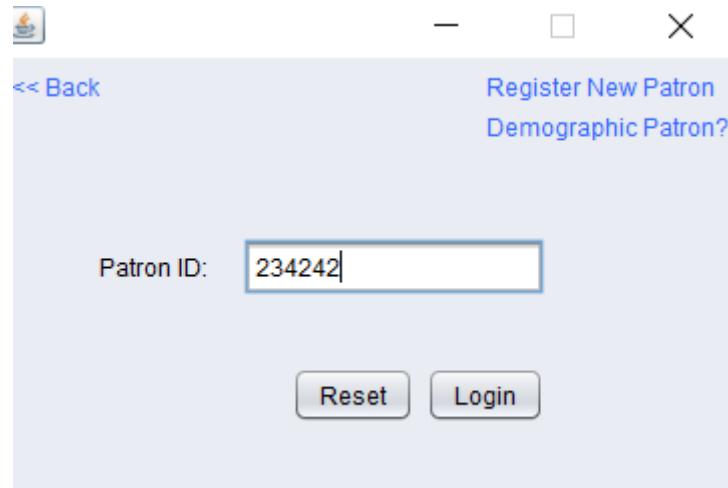
path#1:1-2



Screenshot 6.10

Test NO	Patron_id	Result
1	#2324	See Screenshot 6.10

path#2:1-3-4-7-10



This screenshot shows a user profile page. At the top right are standard window controls for minimize, maximize, and close. Below them is a blue header bar with a logo on the left and a "Logout" link on the right. The main content area displays the following user information:

First Name:	appala
Last Name:	chekuri
Date Of Birth:	1992-10-10
SEX:	Male
EMAIL ADDRESS:	appala.chekuri@sdstate.edu
Phone:	13213128431
Payment:	Not Done
Start Date:	Dec 4, 2016 12:00:00 AM
End Date:	Dec 4, 2016 12:00:00 AM
Access:	Yes

At the bottom center is a blue rectangular button labeled "CHECK-IN".

Screenshot 6.11

Test NO	Patron ID	Result
1	234242	Screenshot 6.11

path#3:1-3-5-8-10

This screenshot shows a user profile page, similar to Screenshot 6.11, but with some additional fields. At the top right are standard window controls. Below them is a blue header bar with a logo on the left and a "Logout" link on the right. The main content area displays the following user information:

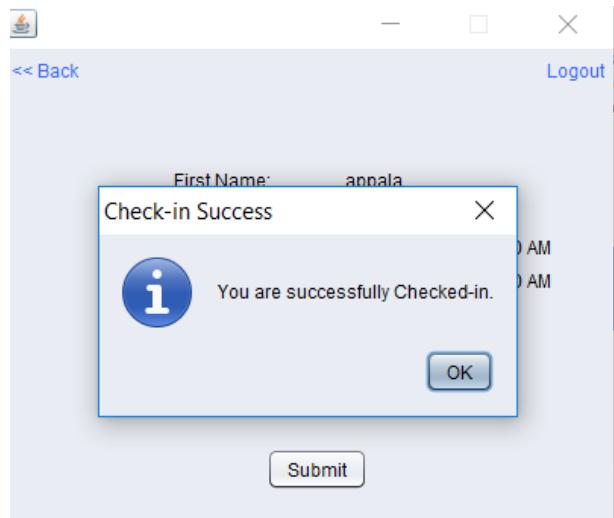
First Name:	appala
Last Name:	chekuri
Start Date:	Dec 4, 2016 12:00:00 AM
End Date:	Dec 4, 2016 12:00:00 AM
Walvor Filled:	Not Done
Payment:	Not Done
Status:	Yes

At the bottom center is a blue rectangular button labeled "Submit".

Screenshot 6.12

Test NO	input	Result
1	submit	See Screenshot 6.12

path#4:1-3-6-9-10



Screenshot 6.13

Test NO	input	Result
1	Click submit	See Screenshot 6.13

Screenshot 6.13

path#5:1-10

The screenshot shows a registration form window with the following fields:

- First Name: [Text input field]
- Last Name: [Text input field]
- User Name: [Text input field]
- Password: [Text input field]
- Confirm Password: [Text input field]
- Email: [Text input field]
- Phone: [Text input field]
- Semester: [Dropdown menu set to Fall]
- Sex: [Dropdown menu set to Male]
- Date Of Birth: [Text input field]

Below the Date Of Birth field is a note: "Format (YYYY-MM-DD)". At the bottom are two buttons: "Reset" and "Submit".

See Screenshot 6.14

Test NO	input	Result
1	Register new patron	See Screenshot 6.14

## 7. INTEGRATION

Login test

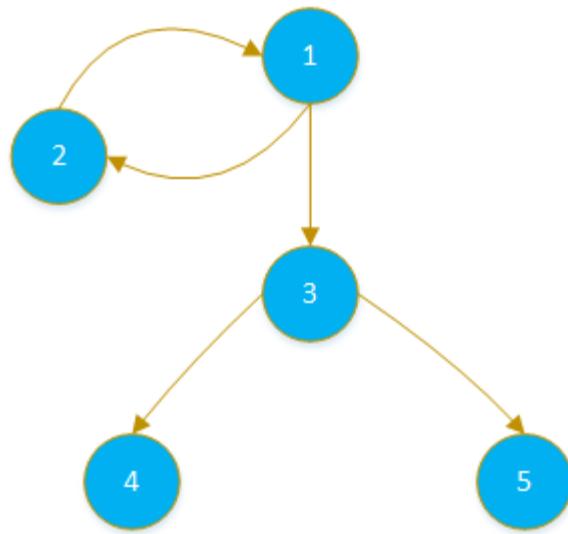
<i>WBS #</i>	<i>Tester</i>	<i>Test Location</i>	<i>Test Date</i>	<i>Developer</i>
1.1	Shaohu	Nester center	12/04/2016	Appala

### Objective:

The employee logins e-Climbing system, and then does check in and registration for patron. We will use top down test approach.

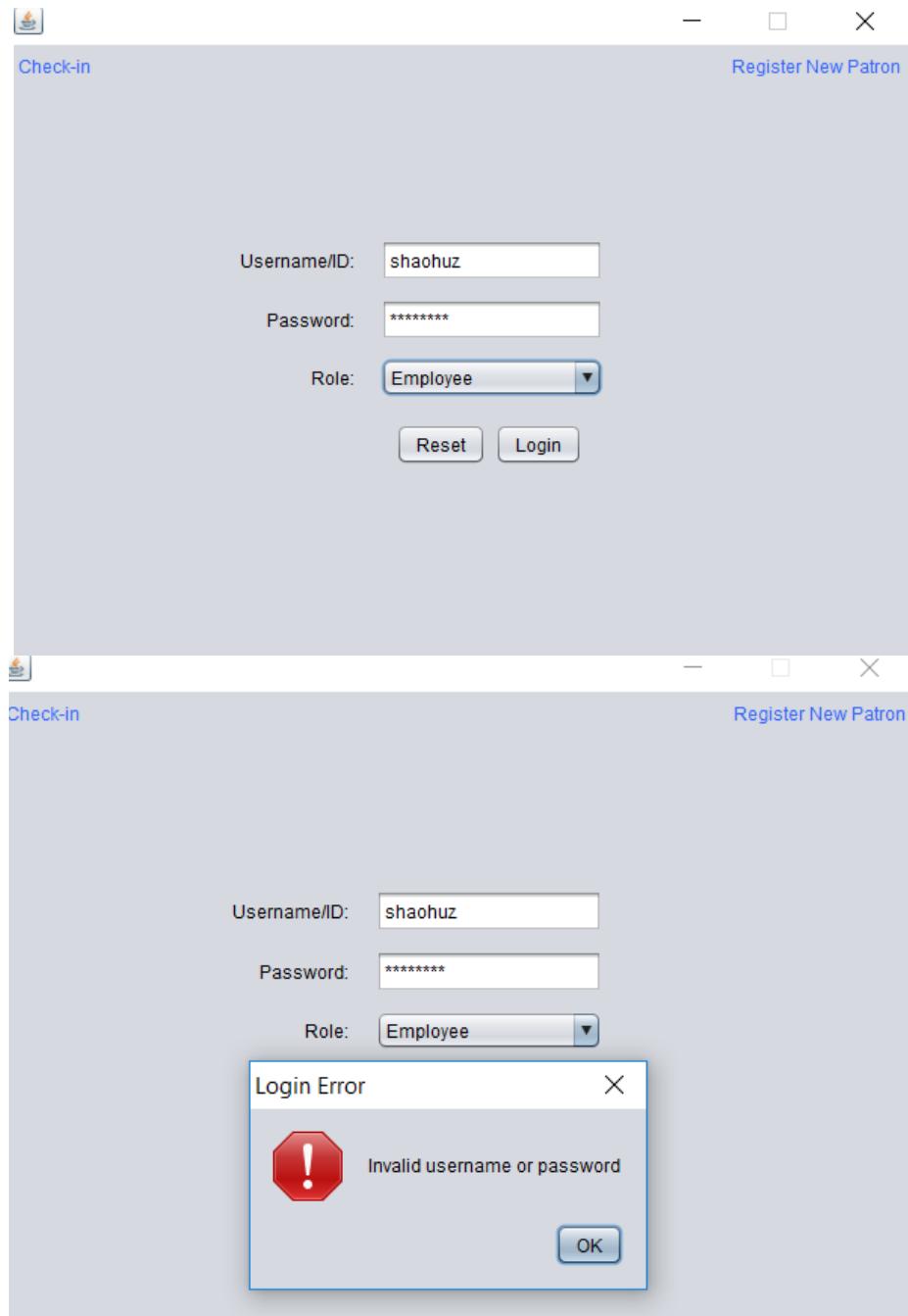
<b>Pseudo code</b>	<pre> if username=valid username and password=valid password     and log_in_button=yes     if Patrin_ID is true         Check in     else         Registration else     print "please enter valid username and password " </pre>
--------------------	--

## 7.1 McCabe's Complexity Graph



## 7.2 Principal path

Path #1:1-2  
Login unsuccessful



Screenshot 7.1

Test NO	input	Result
1	User name: shaohuz Password:12345678	See Screenshot7.1

 Check-in

Register New Patron

Username/ID:

Password:

Role:

 Daily Notes

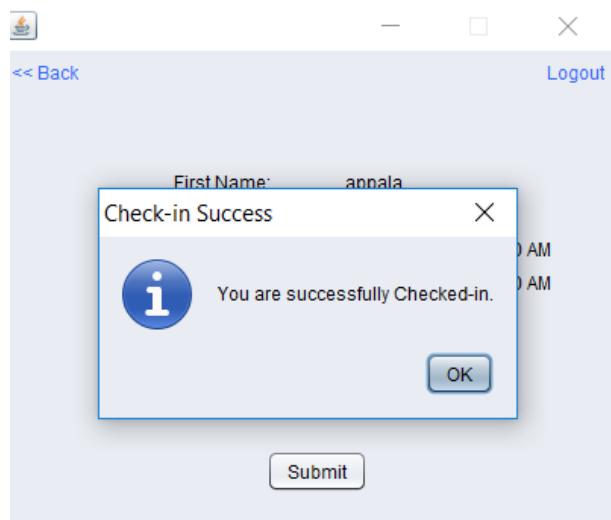
Suspension  
Reservation  
Class Registration  
Patrons  
Create Class  
Update Schedule  
Renew Account

Note:

Send To:

 << Back Register New Patron  
Demographic Patron?

Patron ID:



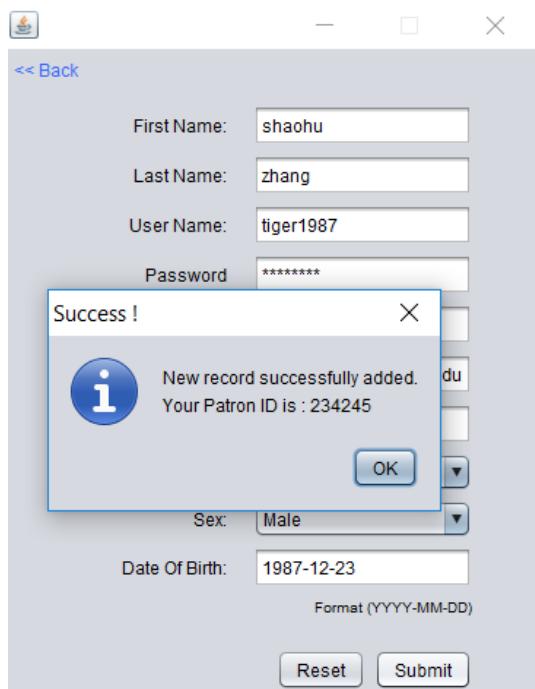
Screenshot 7.2

Path #2:1-3-4

Test NO	input	Result
1	User name: shaohuzhang Password:12345678 Patron ID:234242	See Screenshot 7.2

Path #3:1-3-5

A screenshot of a registration form. The form includes fields for First Name, Last Name, User Name, Password, Confirm Password, Email, Phone, Semester (with 'Fall' selected), Sex (with 'Male' selected), and Date Of Birth. There is also a note about the date format: 'Format (YYYY-MM-DD)'. At the bottom, there are 'Reset' and 'Submit' buttons.



Screenshot 7.3

Test NO	input	Result
1	User name: shaohuzhang Password:12345678 Click registration First Name: shaohu Last Name: zhang Username:tiger1987 Password:12345679 Conform passwaord:1234579 Email: <a href="mailto:shaohu.zhang@sdstate.edu">shaohu.zhang@sdstate.edu</a> Date of birth: 1987-12-23	See Screenshot 7.3

### 7.3 Compound condition

*if username=valid username and password=valid password  
and log in button=yes*

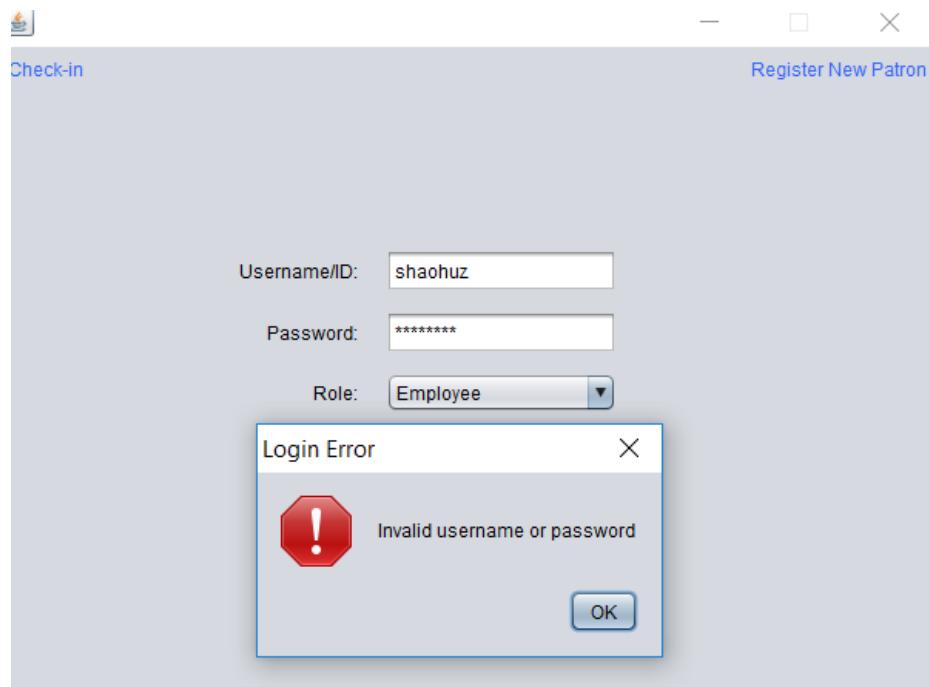
Test No	Username	Password	Result
1	Shaohu	12345678	See screenshot 7.3.1
2	@shaho	12345678	See screenshot 7.3.2

3	Shaohu	#34333	See screenshot 7.3.2
4	@shaho	#34333	See screenshot 7.3.2 and 7.3.3

Test NO	Patron_id	Result
1	232424	See Screenshot 7.3.4
2	@54566	See Screenshot 7.3.5



Screenshot 7.3.1



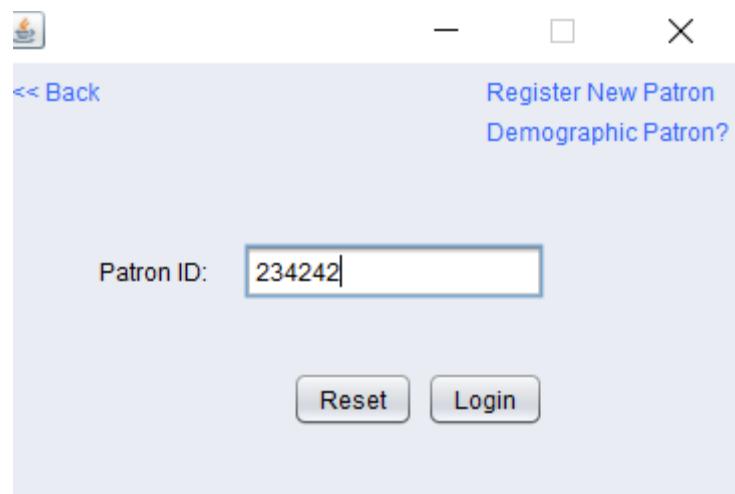
Screenshot 7.3.2

Screenshot 7.3.3 shows a registration form titled "Register New Patron". The form includes the following fields: First Name (text box), Last Name (text box), User Name (text box), Password (text box), Confirm Password (text box), Email (text box), Phone (text box), Semester (dropdown menu set to "Fall"), Sex (dropdown menu set to "Male"), Date Of Birth (text box), and a note "Format (YYYY-MM-DD)". At the bottom are "Reset" and "Submit" buttons.

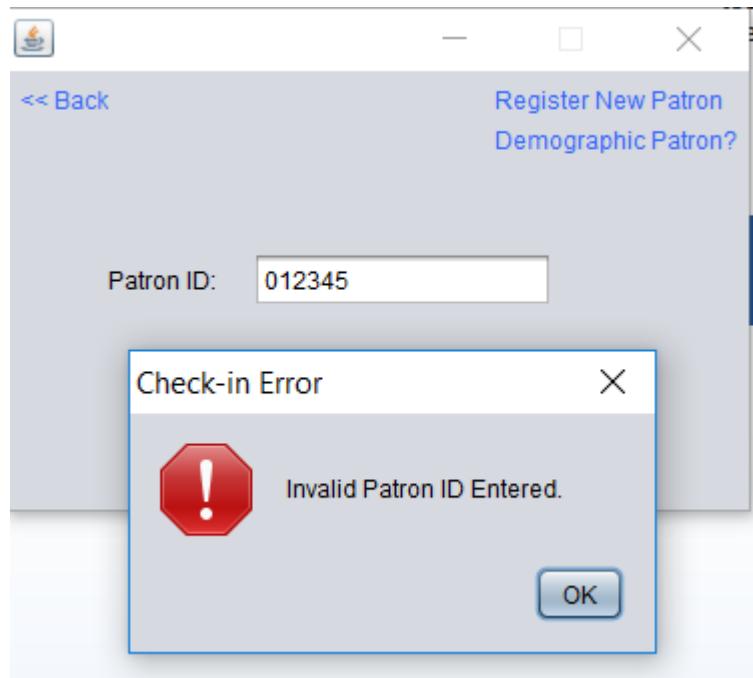
First Name:	<input type="text"/>
Last Name:	<input type="text"/>
User Name:	<input type="text"/>
Password:	<input type="password"/>
Confirm Password:	<input type="text"/>
Email:	<input type="text"/>
Phone:	<input type="text"/>
Semester:	<input type="button" value="Fall"/>
Sex:	<input type="button" value="Male"/>
Date Of Birth:	<input type="text"/>

Format (YYYY-MM-DD)

Screenshot 7.3.3



Screenshot 7.3.4



Screenshot 7.3.5

## 7.4 Domain test

Test No	User name	Password	Result
1	appalaa	r3	Error:invalid username or password
2	ala	1r11!!	Error:invalid username or password
3	pala	!!!r!!!!!!!!!!!!!!111	Error:invalid username or password
4	apala	!!r!!	Error:invalid username or password
5	apala	5454546w53	Error:invalid username or password
6	alaa	r3	Error:invalid username or password
7	algfga	1r11!!	Error:invalid username or password
8	pafgfla	!!!r!!!!!!!!!!!!!!111	Error:invalid username or password
9	apafgla	!!r!!	Error:invalid username or password
10	apcvala	5454546w53	Error:invalid username or password

Comments:

---

---

---

---