BlinkID

Software Requirements Specifications (SRS)

Version 3.0

Document Preparation

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Approval (Signature)** | **Approval Date** |
| Sriharsha Lanka | Developer | Sriharsha Lanka | 04/22/24 |
| Dheeraj Chigurupati | Test Engineer | Dheeraj Chigurupati | 04/22/24 |

Document Approvals

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Approval (Signature)** | **Approval Date** |
| Vyshnavi Balabhadruni | Project Design Manager | Vyshnavi Balabhadruni | 04/22/24 |
| Akash Rao Sudati | Project Configuration Manager | Akash Rao Sudati | 04/22/24 |
| Shiva Kumar Goud Shankaragari | Project Test Manager | Shiva Kumar Goud Shankaragari | 04/22/24 |
| Mounika Reddy Dubba | Project Risk Manager | Mounika Reddy Dubba | 04/22/24 |
| Venkata Sai Ramya Padmasri Boggaram | Project Technical Manager | Venkata Sai Ramya Padmasri Boggaram | 04/22/24 |
| Varun Vanaparthi | Project Manager | Varun Vanaparthi | 04/22/24 |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 23/02/2024 | 1.0 | The first draft | Sriharsha Lanka, Dheeraj Chigurupati |
| 09/04/2024 | 2.0 | Added new use cases and modified use cases | Ritheesh Reddy Gavva,  Chethan Reddy Mallu |
| 22/04/2024 | 3.0 | Made some modifications in all the use cases, requirements, use case diagram | Venkata Sai Ramya Padmasri Boggaram |

Table of Contents

1. [Introduction](#_Introduction) 5

1.1 [Purpose of the Document](#_Purpose_of_the) 5

1.2 [Scope of the Document](#_Scope_of_the) 5

1.3 [References](#_References) 5

1.4 [Definitions, Acronyms, and Abbreviations](#_Definitions,_Acronyms,_and) 6

2. [Product Scope](#_Product_Scope) 6

3. [Product Users](#_Product_Users) 6

3.1 [Users](#_Users) 6

4. [Use Cases](#_Use_Cases) 7

5. [Functional Requirements](#_Functional_Requirements) 7

5.1 [Software Requirements](#_Software_Requirements) 7

5.2 [Hardware Requirements](#_Hardware_Requirements) 8

6. [Quality Requirements](#_Quality_Requirements) 8

7. [Appendix A – Requirements Traceability](#_Appendix_A_–) 9

8. [Appendix B](#_Appendix_B_–) 11

8.1 [University Management](#_Use_Case_1) 11

8.2 [Make a Registration](#_Use_Case_2) 12

8.3 [Creating exams](#_Use_Case_3) 13

8.4 [Enroll students to exams](#_Use_Case_4) 15

8.5 [Validate students in exams](#_Use_Case_5) 16

8.6 [Creating groups](#_Use_Case_6) 17

8.7 [Enroll students to groups](#_Use_Case_7) 18

8.8 [Validate students in groups](#_Use_Case_8) 19

9. [Use Case Diagram](#_Use_Case_Diagram) 21

Software Requirements Document

# Introduction

The functionality, performance, and design requirements for the BlinkID application are described in the software requirements specifications (SRS) document. This document is essential to making sure the project is successful.

## Purpose of the Document

For the BlinkID mobile application, the SRS includes detailed information about the application's biometric authentication and recognition features, such as the ability to authenticate students using facial recognition, track their presence in various campus locations, and provide secure access. It should also include information about the user interface, security features, and integration with other systems, such as student databases and access control systems.

## Scope of the Document

The BlinkID mobile application's software requirements specification (SRS) encompasses all facets of the biometric authentication and recognition features of the application. This covers technical specifications including compatibility with various platforms and operating systems, programming languages, and hardware needs for facial recognition equipment.

User experience (UX) and user interface (UI) design specifications are among the non-technical requirements for BlinkID, assuring the application's intuitiveness and ease of use for administrators and students alike. It should also follow branding standards to provide a unified user experience. The SRS also describes any legal or regulatory obligations pertaining to the processing of biometric data and privacy.

The SRS also identifies BlinkID's performance, scalability, and security requirements to make sure the program can effectively and securely manage the authentication and recognition requirements of a university or college campus.

## References

* Intellij - IntelliJ IDEA, you can debug, analyze, and version the code base of your apps right from within the IDE. It also incorporates the most important developer tools: <https://www.jetbrains.com/idea/>
* Spring Boot - Focus on convention over configuration, Spring Boot is a framework that makes it easier to create production-ready, standalone, and highly scalable Spring-based applications. It also offers a faster method for creating microservices: <https://spring.io/projects/spring-boot>
* VsCode - Microsoft created Visual Studio Code, a lightweight, open-source code editor that is used for a variety of programming languages and activities. Among its many development tools are extensions supported, integrated Git control, and debugging features: <https://code.visualstudio.com/>
* Android Studio - An emulator, a visual designer, and other debugging and testing tools are just a few of the capabilities and tools that Android Studio, a specialized integrated development environment (IDE), offers to speed up the process of developing Android applications: <https://developer.android.com/>
* React.js - A JavaScript framework called React.js is used to create user interfaces. It is especially useful for developing component-based web applications that are dynamic and interactive, since it makes the process of updating and rendering UI elements more straightforward when data changes: <https://react.dev/learn/start-a-new-react-project>
* GitHub - GitHub is a web-based platform that uses Git for version control, enabling collaboration and code sharing among developers. It provides features such as repository hosting, collaboration tools, issue tracking, and continuous integration, making it a widely used platform for software development projects: <https://github.com/>
* Microservices - Software applications are developed and deployed using microservices architecture, which consists of a group of tiny, independent, loosely linked services, each handling a distinct set of business operations: <https://microservices.io/>
* Image classification - Is used for automatically assigning predefined labels or categories to images based on their visual content, finding applications in areas such as object recognition, medical imaging, security, and more: <https://paperswithcode.com/task/image-classification>
* GitLab CI/CD - Is used to build, test, and deploy applications more efficiently by automating the software development lifecycle: <https://about.gitlab.com/solutions/continuous-integration/>
* Project\_smart\_campus\_PPD\_version\_2( <https://drive.google.com/file/d/1k0n1DCK70Ez_OurQhA22LG9MwKlwD7so/view?usp=drive_link>)
* Project\_smart\_campus\_Project\_vision\_Document\_version\_2(<https://drive.google.com/file/d/1sNEibsJLEY_eJ1BWT9CaKe5C81u5NqAZ/view?usp=drive_link>)
* Project\_smart\_campus\_SDD\_version\_2(<https://drive.google.com/file/d/1VtA0Jfs0HiZzkVKC_BNsphXsSO37Zltw/view?usp=drive_link>)
* Project\_smart\_campus\_Software\_Test\_Document\_(STD)\_version\_2(<https://drive.google.com/file/d/1PrSeEJN5LjtrF04M8tCuSCU4CEn7TSg4/view?usp=drive_link>)
* Project\_smart\_campus\_SRS\_version\_2(https://drive.google.com/file/d/1rkHd3ZsuzLRmns8lIaKHoTFJ3oYld3\_4/view?usp=drive\_link)

## Definitions, Acronyms, and Abbreviations

See Appendix A.

# Product Scope

The product scope for BlinkID is to provide a user-friendly platform for implementing student biometric authentication and recognition using facial recognition technology in university or college campuses. This includes providing students, administrators, and security personnel with the tools to manage authentication and verify student presence effectively. The scope of BlinkID includes, but is not limited to: Biometric authentication and recognition, Student presence tracking, User-friendly interface, Integration with campus databases, Security features, Privacy compliance, Performance optimization, and Mobile compatibility.

# Product Users

## Users

|  |  |
| --- | --- |
| **Product User Type** | **User Characteristics** |
| Admin | Age: Administrators are typically between 20-45 years old.  Technology Level: They should have a high level of comfort with technology or possess technical expertise.  Education Level: While a degree in computer science or a related field is preferred, it is not necessarily required.  Other Characteristics: Administrators should have excellent communication skills, strategic thinking abilities, adaptability, and a passion for the causes. |
| Student | Age: Typically college or university students, ranging from late teens to mid-20s, but could vary.  Technology Level: Comfortable with mobile technology and apps, but not necessarily experts.  Education Level: Currently enrolled in a college or university program.  Other Characteristics: Varied interests and backgrounds, likely seeking convenience and efficiency in campus-related tasks. |
| Faculty | Age: Varies, but generally older than students, ranging from late 20s to 60s or older.  Technology Level: Comfortable with technology, but may vary in technical expertise.  Education Level: Advanced degrees (e.g., master's, Ph.D.) in their respective fields.  Other Characteristics: Experienced professionals with a focus on education and research, likely interested in tools that can enhance campus security and efficiency. |
| Staff | Age: Varies, but generally older than students, ranging from late 20s to 60s or older.  Technology Level: Comfortable with technology, but may vary in technical expertise.  Education Level: Advanced degrees (e.g., master's, Ph.D.) in their respective fields.  Other Characteristics: Experienced professionals with a focus on education and research, likely interested in tools that can enhance campus security and efficiency. |

# Use Cases

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | **Brief Description** |
| UC-1 | University Management | This use case involves admins, where admin creates groups within the college for managing access to specific areas such as exams, gym, and library. |
| UC-2 | Make a Registration | This use case involves allowing students/faculty to register their biometric data (facial recognition) with the BlinkID system, including providing personal information and student ID details. |
| UC-3 | Creating exams | This use case involves faculties creating exams as tasks for students where each exam contains set of students who are enrolled for that particular exam. |
| UC-4 | Enroll students to exams | This use case involves enrolling students to created exams by faculty. |
| UC-5 | Validate students in exams | This use case involves validating students to enrolled exams by faculty. |
| UC-6 | Creating groups | This use case involves faculties creating groups for students where each group contains set of students who are enrolled for that particular group. |
| UC-7 | Enroll students to groups | This use case involves enrolling students to created groups by staff. |
| UC-8 | Validate students in groups | This use case involves validating students to enrolled groups by staff. |

# Functional Requirements

## Software Requirements

|  |  |  |
| --- | --- | --- |
| **Requirement ID** | **Requirement Priority**  **(High/Medium/Low)** | **Requirement** |
| Req-Func-BID-1 | High | The system shall allow administrators to create groups within the college for managing access to specific areas such as exams, gym, and library. |
| Req-Func-BID-2 | High | The system shall allow students/faculty to register their biometric data (facial recognition) with the BlinkID system, including providing personal information and student ID details. |
| Req-Func-BID-3 | High | The system shall allow faculty/staff to create specific tasks for the students which are going to be tracked. |
| Req-Func-BID-4 | High | The system shall authenticate students using facial recognition at various campus locations for access control. |
| Req-Func-BID-5 | High | The system shall track student presence in classes, labs, and other academic activities using facial recognition. |
| Req-Func-BID-6 | High | The system shall allow administrators to manage student activity, including tracking student presence. |
| Req-Func-BID-7 | High | The system shall include various verification methods, such as ID verification, to designate a user as a certified user. |
| Req-Func-BID-8 | High | The system shall provide tools for administrators to track and analyze student biometric authentication and recognition data, authentication success rates, and system performance. |
| Req-Func-BID-9 | High | The system shall employ BlinkID to enable security personnel to verify that only authorized individuals have access to specific areas of the campus, thereby enhancing overall campus security. |
| Req-Func-BID-10 | High | The system shall provide a user-friendly interface for administrators to manage user groups and access permissions. |
| Req-Func-BID-11 | Medium | The system shall allow administrators to easily add, edit, or remove tasks for students. |
| Req-Func-BID-12 | Low | The system shall provide real-time notifications to students regarding new tasks, task updates, and approaching deadlines. |
| Req-Func-BID-13 | Low | The system shall provide detailed logs of user activity, including login/logout times, task completion status, and system interactions. |
| Req-Func-BID-14 | Medium | The system shall allow students to view their task assignments, deadlines, and completion status through a user-friendly dashboard. |
| Req-Func-BID-15 | Medium | The system shall integrate with existing student information systems to retrieve and update student data for authentication and tracking purposes. |

## Hardware Requirements

|  |  |  |
| --- | --- | --- |
| **Requirement ID** | **Requirement Priority**  **(**High/Medium/Low**)** | **Requirement** |
| Req-Func-Hw-1 | High | The system shall be designed to be compatible with a wide range of devices which includes smart phones and tablets. |
| Req-Func-Hw-2 | High | The system shall have a visible camera. |
| Req-Func-Hw-3 | Medium | The system shall have a stable internet connection to access the application. |
| Req-Func-Hw-4 | High | The system must adhere to all applicable hardware, software, and standards and laws in order to guarantee compatibility and security. |
| Req-Func-Hw-5 | High | The system shall ensure a good user experience. |

# Quality Requirements

|  |  |  |
| --- | --- | --- |
| **Requirement ID** | **Requirement Priority**  **(**High/Medium/Low**)** | **Requirement** |
| Req-Qual-1 | High | The system shall have a simple architecture and be easy to use. |
| Req-Qual-2 | High | The system shall not allow unauthorized access to the system. |
| Req-Qual-3 | Medium | The system shall be easily maintainable by securing and protecting user data. |
| Req-Qual-4 | High | The system shall have high performance with fast response and minimum failures. |
| Req-Qual-5 | High | The system shall be reliable and available for all at all the times. |

# Appendix A – Requirements Traceability

|  |  |
| --- | --- |
| **Test Case ID** | **Requirement ID** |
| TC-1 | Req-Func-BID-1  Req-Func-Hw-2  Req-Qual-4  Req-Qual-5 |
| TC-2 | Req-Func-BID-2  Req-Func-BID-3  Req-Func-BID-1  Req-Func-BID-2  Req-Qual-Hw-5  Req-Qual-1  Req-Qual-4 |
| TC-3 | Req-Func-BID-9  Req-Func-BID-10  Req-Func-Hw-3  Req-Func-Hw-5  Req-Qual-2  Req-Qual-3 |
| TC-4 | Req-Func-BID-4  Req-Func-Hw-5  Req-Qual-1 |
| TC-5 | Req-Func-BID-6  Req-Func-BID-7  Req-Qual-Hw-5  Req-Qual-3 |
| TC-6 | Req-Func-BID-2  Req-Func-BID-12  Req-Qual-Hw-3  Req-Qual-2 |
| TC-7 | Req-Func-BID-5  Req-Func-BID-11  Req-Qual-Hw-3  Req-Qual-3 |
| TC-8 | Req-Func-BID-4  Req-Func-BID-10  Req-Qual-Hw-1  Req-Qual-4 |
| TC-9 | Req-Func-BID-8  Req-Func-BID-3  Req-Qual-Hw-4  Req-Qual-3 |
| TC-10 | Req-Func-BID-9  Req-Func-BID-4  Req-Qual-Hw-2  Req-Qual-1 |
| TC-11 | Req-Func-BID-2  Req-Func-BID-7  Req-Qual-Hw-1  Req-Qual-3 |
| TC-12 | Req-Func-BID-6  Req-Func-BID-3  Req-Qual-Hw-4  Req-Qual-2 |
| TC-13 | Req-Func-BID-4  Req-Func-BID-1  Req-Qual-Hw-1  Req-Qual-5 |
| TC-14 | Req-Func-BID-3  Req-Func-BID-9  Req-Qual-Hw-3  Req-Qual-4 |
| TC-15 | Req-Func-BID-5  Req-Func-BID-8  Req-Qual-Hw-2  Req-Qual-2 |
| TC-16 | Req-Func-BID-6  Req-Func-BID-2  Req-Qual-Hw-1  Req-Qual-1 |
| TC-17 | Req-Func-BID-5  Req-Func-BID-1  Req-Qual-Hw-4  Req-Qual-5 |
| TC-18 | Req-Func-BID-3  Req-Func-BID-11  Req-Qual-Hw-1  Req-Qual-2 |

# Appendix B –

# Use Case 1 – University Management

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | UC-1 | |
| **Use Case Name** | University Management | |
| **Brief Description** | This use case involves admins creating groups within the college for managing access to specific areas such as exams, gym, and library. | |
| **Frequency of Use** | High | |
| **Priority** | High | |
| **Current Version** | 2.0 | |
| **Date of First Version** | 02/23/2024 | |
| **Date of Last Version** | 04/08/2024 | |
| **Created By** | Chethan Reddy Mallu | |
| **Last Update By** | Chethan Reddy Mallu | |
| **Approved By** | Varun Vanaparthi | |
| **Assumptions** | The system is fully operational, and admins have the necessary permissions to create and manage groups. | |
| **Primary Actor** | Admin | |
| **Secondary Actor/s** | None | |
| **Preconditions** | The admin is logged into the system and has the necessary permissions to create groups. | |
| **Postconditions** | The groups are successfully created and can be used to manage access to specific areas. | |
| **Trigger** | The admin opens the admin panel and selects the option to create a new group. | |
| **Main Success Scenario** | **User Actions** | **System Actions** |
|  | 1. The admin selects the option “Add Group”.  3. The admin enters the group's name, description and clicks on “Add Group” button.  5. The admin confirms the creation of the group by checking it in Groups. | 2. The system displays a form for the admin to enter the group's details, such as name and description.  4. The system creates the group that admin created and the created group is added to groups list.  . |
| **Alternate Scenarios** | **User Actions** | **System Actions** |
|  |  |  |
| **Additional Notes (Constraints, etc.)** |  | |

# Use Case 2 – Make a Registration

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | UC-2 | |
| **Use Case Name** | Make a Registration | |
| **Brief Description** | This use case describes about the process of registering a new student on the BlinkID mobile application | |
| **Frequency of Use** | High | |
| **Priority** | High | |
| **Current Version** | 2.0 | |
| **Date of First Version** | 02/23/2024 | |
| **Date of Last Version** | 04/07/2024 | |
| **Created By** | Ritheesh Reddy Gavva | |
| **Last Update By** | Ritheesh Reddy Gavva | |
| **Approved By** | Varun Vanaparthi | |
| **Assumptions** | The system is in fully working condition where admin will create an register for the individuals. | |
| **Primary Actor** | Admin | |
| **Secondary Actor/s** | Staff, Teacher | |
| **Preconditions** | The admin is logged into the system. | |
| **Postconditions** | The individual’s register information needs to be stored in the database | |
| **Trigger** | The admin opens up the app and selects the “Register” option to register new students. | |
| **Main Success Scenario** | **User Actions** | **System Actions** |
|  | 1. The admin will go to “Teachers” button and will click on it.  3. The admin enters teacher’s email address, user name, and other required information in the registration form along with picture and clicks “Add Teacher”.  5. The admin will go to “staff” button and will click on it.  7. The admin enters staff’s email address, user name, and other required information in the registration form along with picture and clicks “Add Staff”.  9. The admin will go to “Students” button and will click on it.  11. The admin enters student’s email address, user name, and other required information in the registration form along with picture and clicks “Add Student”. | 2. The system displays the list of teacher’s present along with the “Add New” button.  4. The system will add the new teacher to existing list of teachers.  6. The system displays the list of staff present along with the “Add New” button.  8. The system will add the new staff to existing list of staff.  10. The system displays the list of student’s present along with the “Add New” button.  12. The system will add the new student to existing list of students. |
| **Alternate Scenarios** | **User Actions** | **System Actions** |
|  |  |  |
| **Additional Notes (Constraints, etc.)** |  | |

# Use Case 3 – Creating Exams

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | UC-3 | |
| **Use Case Name** | Creating exams | |
| **Brief Description** | This use case involves teachers creating exams as tasks for students where each exam contains set of students who are enrolled for that particular exam. | |
| **Frequency of Use** | High | |
| **Priority** | High | |
| **Current Version** | 2.0 | |
| **Date of First Version** | 02/21/2024 | |
| **Date of Last Version** | 04/05/2024 | |
| **Created By** | Sai Charan Kammampally | |
| **Last Update By** | Sai Charan Kammampally | |
| **Approved By** | Varun Vanaparthi | |
| **Assumptions** | The system is fully operational, and teachers have the necessary permissions to create and manage exams for students. | |
| **Primary Actor** | Teacher | |
| **Secondary Actor/s** | Admin | |
| **Preconditions** | The teacher is logged into the system and has the necessary permissions to create exams. | |
| **Postconditions** | The exams are successfully created and can be used to enroll specific students. | |
| **Trigger** | The teacher opens the teacher panel and selects the option to create a new exam. | |
| **Main Success Scenario** | **User Actions** | **System Actions** |
|  | 1.The teacher is logged in using username and password in the login page.  3. The teacher selects the option to create a new exam called “add exam”.  5. The teacher clicks on “add exam” button to create the exam.  7. The teacher confirms the creation of the exams. | 2. The system displays “login successful” message.  4. The system displays a form for the teacher to enter the exam details, such as name, description, time, date, location and duration.  6. The system shows exam added successfully screen.  8. In the “exams” the list of exams created are shown.  . |
| **Alternate Scenarios** | **User Actions** | **System Actions** |
|  | 1.a. The teacher enters invalid login information. | 2.a. The system displays an error message and prompts the faculty to correct the login details. |
| **Additional Notes (Constraints, etc.)** |  | |

# Use Case 4 – Enroll students to exams

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | UC-4 | |
| **Use Case Name** | Enroll students to tasks | |
| **Brief Description** | This use case involves enrolling students to created exams by teacher. | |
| **Frequency of Use** | High | |
| **Priority** | High | |
| **Current Version** | 2.0 | |
| **Date of First Version** | 02/21/2024 | |
| **Date of Last Version** | 04/05/2024 | |
| **Created By** | Sriharsha Lanka | |
| **Last Update By** | Ritheesh Reddy Gavva | |
| **Approved By** | Varun Vanaparthi | |
| **Assumptions** | The system is fully operational, and teachers have the necessary permissions to enroll students to created exams. | |
| **Primary Actor** | Teacher | |
| **Secondary Actor/s** | None | |
| **Preconditions** | The teacher is logged into the system and has the necessary exam created to enroll students. | |
| **Postconditions** | The teacher successfully enrolled students to the exam. | |
| **Trigger** | The teacher opens the teacher panel and selects the option to exams. | |
| **Main Success Scenario** | **User Actions** | **System Actions** |
|  | 1. The teacher selects the option “exams” button.  3. Click on the addition button to enroll students.  5. The search bar is used to search particular student using name.  7. Select the students you want to enroll. | 2. The system displays exam details along with invigilators and bottom you will find delete and addition buttons.  4. The system shows list of students that teacher can enroll.  6. The particular student name is popped up as a result.  8. The selected student names are resulted in the enrolled students list.  . |
| **Alternate Scenarios** | **User Actions** | **System Actions** |
|  |  |  |
| **Additional Notes (Constraints, etc.)** |  | |

# Use Case 5 – Validate students in exams

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | UC-5 | |
| **Use Case Name** | Validate students in exams | |
| **Brief Description** | This use case involves validating students to enrolled exams by teacher. | |
| **Frequency of Use** | High | |
| **Priority** | High | |
| **Current Version** | 2.0 | |
| **Date of First Version** | 02/21/2024 | |
| **Date of Last Version** | 04/10/2024 | |
| **Created By** | Ritheesh Reddy Gavva | |
| **Last Update By** | Sai Charan Kammampally | |
| **Approved By** | Varun Vanaparthi | |
| **Assumptions** | The system is fully operational, and teachers can validate enrolled students. | |
| **Primary Actor** | Teacher | |
| **Secondary Actor/s** | None | |
| **Preconditions** | The teacher is logged into the system and has the necessary permissions to validate students. | |
| **Postconditions** | The teacher successfully validates students. | |
| **Trigger** | The teacher opens the teacher panel and selects the option to exams. | |
| **Main Success Scenario** | **User Actions** | **System Actions** |
|  | 1. The teacher selects the option “exams”.  3. Click on the particular student you want to validate.  5. Click on the validate student button.  7. click the photo of student you selected to validate. | 2. The system displays exam details along with invigilators and bottom you will find delete and addition buttons.  4. The system shows exam description of student enrolled and a button with validate student.  6. Camera gets opened.  8. Student gets validated with success message.  . |
| **Alternate Scenarios** | **User Actions** | **System Actions** |
|  |  | 8.a. Student’s validation failed with error message. |
| **Additional Notes (Constraints, etc.)** |  | |

# Use Case 6 – Creating Groups

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | UC-6 | |
| **Use Case Name** | Creating groups | |
| **Brief Description** | This use case involves staff creating groups for students where each group contains set of students who are enrolled for that particular group. | |
| **Frequency of Use** | High | |
| **Priority** | High | |
| **Current Version** | 2.0 | |
| **Date of First Version** | 02/21/2024 | |
| **Date of Last Version** | 04/05/2024 | |
| **Created By** | Venkata Sai Ramya Padmasri Boggaram | |
| **Last Update By** | Sriharsha Lanka | |
| **Approved By** | Varun Vanaparthi | |
| **Assumptions** | The system is fully operational, and faculties have the necessary permissions to create and manage groups for students. | |
| **Primary Actor** | Staff | |
| **Secondary Actor/s** | None | |
| **Preconditions** | The staff is logged into the system and has the necessary permissions to create groups. | |
| **Postconditions** | The groups are successfully created and can be used to enroll specific students. | |
| **Trigger** | The staff opens the staff panel and selects the option to create a new group. | |
| **Main Success Scenario** | **User Actions** | **System Actions** |
|  | 1.The staff is logged in using username and password in the login page.  3. The staff selects the option to create a new exam called “add group”.  5. The staff clicks on “add group” button to create the group.  7. The staff confirms the creation of the groups. | 2. The system displays “login successful” message.  4. The system displays a form for the staff to enter the name and description of the group.  6. The system shows group added successfully screen.  8. In the groups the list of groups created are shown.  . |
| **Alternate Scenarios** | **User Actions** | **System Actions** |
|  | 1.a. The staff enters invalid login information. | 2.a. The system displays an error message and prompts the staff to correct the login details. |
| **Additional Notes (Constraints, etc.)** |  | |

# Use Case 7 – Enroll students to groups

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | UC-7 | |
| **Use Case Name** | Enroll students to groups | |
| **Brief Description** | This use case involves enrolling students to created groups by staff. | |
| **Frequency of Use** | High – 100% | |
| **Priority** | High | |
| **Current Version** | 2.0 | |
| **Date of First Version** | 02/21/2024 | |
| **Date of Last Version** | 04/05/2024 | |
| **Created By** | Sriharsha Lanka | |
| **Last Update By** | Mounika Dubba | |
| **Approved By** | Varun Vanaparthi | |
| **Assumptions** | The system is fully operational, and staff have the necessary permissions to enroll students to created groups. | |
| **Primary Actor** | Staff | |
| **Secondary Actor/s** | None | |
| **Preconditions** | The staff is logged into the system and has the necessary group created to enroll students. | |
| **Postconditions** | The staff successfully enrolled students to the group. | |
| **Trigger** | The staff opens the staff panel and selects the option to groups. | |
| **Main Success Scenario** | **User Actions** | **System Actions** |
|  | 1. The staff selects the option “groups”.  3. Click on the addition button to enroll students.  5. The search bar is used to search particular student using name.  7. Select the students you want to enroll. | 2. The system displays exam details along with invigilators and bottom you will find delete and addition buttons.  4. The system shows list of students that staff can enroll.  6. The particular student name is popped up as a result.  8. The selected student names are resulted in the enrolled students list. |
| **Alternate Scenarios** | **User Actions** | **System Actions** |
|  |  |  |
| **Additional Notes (Constraints, etc.)** |  | |

# Use Case 8 – Validate students in groups

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | UC-8 | |
| **Use Case Name** | Validate students in groups | |
| **Brief Description** | This use case involves validating students to enrolled groups by staff. | |
| **Frequency of Use** | High – 100% | |
| **Priority** | High | |
| **Current Version** | 2.0 | |
| **Date of First Version** | 02/21/2024 | |
| **Date of Last Version** | 04/10/2024 | |
| **Created By** | Varun Vanaparthi | |
| **Last Update By** | Chethan Reddy Mallu | |
| **Approved By** | Varun Vanaparthi | |
| **Assumptions** | The system is fully operational, and staff can validate enrolled students. | |
| **Primary Actor** | Staff | |
| **Secondary Actor/s** | None | |
| **Preconditions** | The staff is logged into the system and has the necessary permissions to validate students. | |
| **Postconditions** | The staff successfully validates students. | |
| **Trigger** | The staff opens the staff panel and selects the option to groups. | |
| **Main Success Scenario** | **User Actions** | **System Actions** |
|  | 1. The staff selects the option “groups”.  3. Click on the particular student you want to validate.  5. Click on the validate student button.  7. click the photo of student you selected to validate. | 2. The system displays group details along with invigilators and bottom you will find delete and addition buttons.  4. The system shows group description of student enrolled and a button with validate student.  6. Camera gets opened.  8. Student gets validated with success message  . |
| **Alternate Scenarios** | **User Actions** | **System Actions** |
|  |  | 8.a. Student’s validation failed with error message. |
| **Additional Notes (Constraints, etc.)** |  | |

# Use Case Diagram

A diagram of a student's process

Description automatically generated