Appendix

FORM 1



**FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY**

**Diploma in Software Engineering**

**Programme: DSF (Group: ONE)**

**Assignment**

## **AMSE1003 SOFTWARE ENGINEERING**

| Name (Block Letters) | Registration No. | Signature | Marks |
| --- | --- | --- | --- |
| 1. ABIGAIL AIMEE LATOJA | 25SMD05881 |  |  |
| 2. IVAN FOO JIA HAO | 25SMD07087 |  |  |
| 3. IRLEESYA ERZA BINTI RUNZEE @ SYAMREE | 25SMD05589 |  |  |
| 4. LIM SUK YI | 25SMD05847 |  |  |
| 5. ADAM YONG ZHENG QUAN | 25SMD01932 |  |  |

Lecturer’s Name: Surayaini Binti Basri

Date of Submission: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

FORM 2



**FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY**

**Plagiarism Statement and Guideline for Late Submission of Coursework**

**Read, complete, and sign this statement to be submitted with the written report.**

**We confirm that the submitted works are all our own work and are in our own words.**

| Name (Block Letters) | Registration No. | Signature | Date |
| --- | --- | --- | --- |
| 1. AIMEE LATOJA ABIGAIL | 25SMD05881 |  | 21th September 2025 |
| 2. IVAN FOO JIA HAO | 25SMD07087 |  | 21th September 2025 |
| 3. IRLEESYA ERZA BINTI RUNZEE @ SYAMREE | 25SMD05589 |  | 21th September 2025 |
| 4. LIM SUK YI | 25SMD05847 |  | 21th September 2025 |
| 5. ADAM YONG ZHENG QUAN | 25SMD01932 |  | 21th September 2025 |

# **UMS Hostel Attendance System**

## **Problems of Existing System**

1. **Location:** Jalan UMS, Kota Kinabalu, Sabah.
2. **Objective:**
3. To strengthen the hostel attendance system.
4. **Problems with the manual hostel attendance system:**
5. Inefficient in storage:

i) When searching for information, it can potentially hinder productivity due to time consumption.

ii) Manually taking attendance needs a lot of time, especially in large hostels.

iii) No backup exists if records are lost.

1. Information may not be stored properly, causing losses:

i) May be caused by human error such as accidental deletion or mishandling of data.

ii) Information recorded in the manual hostel attendance system, which is written in a book, the book may be damaged because of water, fire, tearing, or other accidents that may occur.

iii) The book used to record the attendance could be misplaced causing overall losses, unless the book is found.

1. Inaccurate information:

i) Mistakes in spelling might go unnoticed, causing invalid information.

ii) Students may tamper with the record without detection leading to severe data integrity issues.

1. Unclear handwriting:

i) Because of the difference in human handwriting style, some characters may be difficult to read.

ii) Slows down the process when writing down attendance, causing time consumption.

iii) In emergencies, unclear handwriting makes it hard to identify who’s present or missing, risking delays and safety.

1. Possibility of information leakage:

i) Students might leak confidential information because of the unsecure security system of the software which could lead to impersonation, misuses of information, and more.

1. Difficulty in Monitoring Leave and Movement

i) Tracking students who are on leave or permission is harder with paper logs.

ii) Risk of students going out without proper permission.

iii) Students might invite unnecessary guests into the hostel without notice.

# **Software Quality Attributes**

## **Acceptability**

i) The software to be created should be acceptable by students and staff.

ii) It should be very compatible with the system that the students and staff use.

iii) The software should be easy to use without complicated user manuals. //

iv) The software should be easy to use when incapable people use the software.// slow learners we add in later

v) Clear, helpful prompts improve user trust

vi) Able to be customized by the student to fit every student's schedule .

## **Dependability and security**

i) The software should be secure to avoid malicious cyber threats such as malwares and ransomware, causing damage to the system and its users.

ii) A dependable software should not have any failure that can cause physical or economic damage to its users.

iii) User Access to the system should be controlled based on roles and permission, ensuring that only authorised personnel can view or modify data.

iv) Passwords must be written with strong complexity rules and no plain-text storage, to protect user credentials from breaches.

## **Efficiency**

i) The software should not be able to make any wasteful use of the system’s resources, especially the memory and processor cycles. Therefore, it should optimize the use of resources to minimize waste and maximize output.

ii) The processing time should not take more than 2 seconds to avoid web traffic and disrupt users’ productivity when using the system.

iii) The system should be responding well upon users’ interaction.

iv)Resources Should ensure that resources within the system are used for the right task and contribute to the desired outcomes.

## 

## **Maintainability**

i) The software should be written in a way that it can fulfill the needs of its users without taking too much time when evolving the software.

ii) The software must have complete code documentation, database flexibility, clear version control, and a modular code base for a faster maintenance process.

iii) Proper documentation should be provided to help future developers understand, fix, and enhance the system.

iv) The system should have good error logging capabilities to facilitate inspection and repair.

# 

# **Software Process Model**

## **Lean Software Development**

This methodology prioritizes delivering value to customers by minimizing waste and maximizing efficiency in the development process. Key principles of this methodology include eliminating waste, building in quality, fast delivery, respecting people, and optimizing to see the whole. fn

## **Delivering fast of the dormitory record model allows it to be used quickly.**

* If you finish the product early, you can check if the user likes it and fix problems quickly. Also, early delivery helps save time and money.
* Fast delivery can reduce the need to store hand written records and lower the chance of making mistakes.

## **Anything that does not contribute to the project will be labelled as a waste and will be eliminated.**

* Doesn't need to carry out any unnecessary repetition of tasks
* Identifying and removing any activity that does not add value to the end product. (i.e, overproduction, waiting, unnecessary transport, over processing, defects )
* Using a simple UI instead of complicating it.

## **See the whole.**

* This principle means we need to focus on the entire workflow or process, not just the isolated tasks.
* Instead of only focusing on building the attendance form, Developers should also consider how wardens, students, and admins will use and access the data making sure the whole system will run smoothly.

## **Every member of the team should have interdisciplinary cooperation on every step of development.**

* Any existing problems should not be ignored if it’s seen. If it were to be ignored even though its existence was known, bigger problems may occur such as increase in costs, delayed timelines, compromised quality, and reduced productivity.
* Every working member in the team should consider the entire value stream and optimize processes for the overall benefit of the project, not just for individual parts. In contrast, without a shared understanding of the entire process, teams may optimize their individual tasks at the expense of the overall system, leading to increased waste, bottleneck, and difficulty in measuring value.

## **Defer Commitment**

* Delay decisions until you have enough information, This will encourage better decisions based on facts and not guesses .
* Don't finalize the system features too early, instead the developers should wait to understand the hostel’s real needs first, so the design fits them better.

## **Build quality in.**

* This principle means that quality should not be added after development through testing alone but should be built into every step of the development process.
* Developers write clear code and use automated tests from the beginning.For example,bugs like “missing check-in data” or “duplicate entries” are caught immediately before they affect users.

As a conclusion, by embracing the principles and practices of Lean Software Development, organizations can optimize their development processes, deliver higher quality software, and ultimately achieve greater success.

# 

# **Project Plan and Schedule**

## **Task 1: Set project objective**

Identify what students, wardens, and admins need like marking attendance, getting reports, login security.

## **Task 2: Finding and Fixing waste in process**

Analyze current manual attendance issues. For example,paper-based errors, delays and find ways to eliminate waste.

## **Task 3: Check and prevent mistakes early**

Use Test-Driven Development (TDD) to create reliable login and attendance modules from the start.

## **Task 4: Deliver login and attendance**

Create a login system and daily attendance submission core features.

## **Task 5: User feedback and fixes**

Get feedback from test users, fix usability and functionality bugs.

## **Task 6: Add reports and test**

Enable daily or weekly attendance reports useful for staff or admin

## **Task 7: Build admin in dashboard**

## **Task 8: Optimize and refactor**

Improve system performance, make the interface user-friendly, and fix technical issues.

## **Task 9: Final testing and integration**

Ensure all parts operate together smoothly.

## **Task 10: Deploy and monitor**

Install system in real hostel environment.Check for real-time usage, logs, and errors.

## **Task 11: Post-launch fixes and review**

Fix bugs, respond to real user problems and prepare for future improvements.

# 

# **Software Requirements Specification**

## **1.1 Functional requirement**

### **User Authentication**

1.1.1 The system shall allow users to log in using their student id.

1.1.2 The system shall allow different access levels based on roles.

1.1.3 The system can only be accessible by registered students.

1.1.4 The system should automatically log out users after a period of inactivity.

### **Student Registration & Management**

1.1.5 The system shall generate a daily students attendance list including student names.

1.1.6 The system must allow only staff to edit the attendance records.

1.1.7 The system should enable viewing of individual student attendance records.

1.18 The system should allow new students to register by filling in personal details such as name, student ID, contact information, and program of study.

### **Attendance Marking**

1.1.9 The system sends an email to the student, when a student registers their attendance .

1.1.10 The system can generate student attendance in pdf format.

1.1.11 The system should allow students to submit corrections to attendance records with valid reasons for wardens approval.

1.1.12 The system should prevent duplicate attendance entries for the same student in a single session.

### **Attendance Viewing and Reporting**

1.1.13 The system should highlight students whose attendance fails below a set threshold (e.g. 80%).

1.1.14 The system should allow students to view their attendance record for individual subjects or sessions.

1.1.15 The system should include a quick search bar for wardens to find specific students in large class attendance reports.

1.1.16 The system should provide a mobile friendly alternative for quick access to view the students attendance.

### **System Usability**

1.1.17 The system load page and process requests should not take longer than 4 seconds.

1.1.18 The system should provide clear navigation menus with logical grouping of features.

1.1.19 The system should provide a confirmation prompt before performing an irreversible action.

1.1.l.20 The system should include a feedback button for users to report issues or suggest improvements.

## **1.2 Non-functional requirements**

1.2.1 The system shall not disclose any personal information of students.

1.2.1 The system should provide an easy-to-use interface.

1.2.3 The system should be designed to allow for future updates and modifications easily.

1.2.4 The system should be able to handle an increasing number of students and attendance records over time.

1.2.5 Users must change the initially assigned login password immediately after the first successful login. Moreover, the initial should never be reused.

# **Architectural design**

## **Client-server model**

* Each server can be used by multiple clients (students & lectures) at once.
* Easier to add new servers or upgrade to existing servers
* The communication channel such as WiFi, LAN, and internet that connects clients and servers ensures that no matter where the warden or student is, they can interact with the server.
* Enable the clients (student, wardens, admin staff) to access the system through apps, web browser, or kiosks.
* It has a backup server so that when the primary server fails, the secondary server ensures continued operation.
* Provides better reliability since servers can be monitored and managed 24/7.
* It is easier to add new servers or upgrade existing ones to handle growing demands.

# **Test Cases**

| Program Name : UMS Hostel attendance system  Test Date : 12th September 2025 Tester: Irlessya, Abigail, Lim, Adam, Ivan | | | | | |
| --- | --- | --- | --- | --- | --- |
|
| No | Objective/Test Cases | Test Data | Expected Result | Actual Results | Remarks/Comments |
| 001 | To generate a list of hostel student attendance for a selected day | Daily attendance data of hostel students | A list of hostel students with their attendance status (present/absent/late) is displayed for the selected day. |  |  |
| 002 | Verify login with valid Student ID and password | Student ID: 25SMD05881  Password: Test@231 | User successfully logs in and is redirected to the dashboard |  |  |
| 003 | Verify login with invalid credentials | Student ID: 25SMD058881  Password: wrongPassword | System rejects login and shows error message |  |  |
| 004 | Auto logout after a period of inactivity | After login, user stays idle for more than 10 minutes | System automatically logs out with the popup “Session timeout” and shows login screen |  |  |
| 005 | View attendance threshold alerts | Students’ attendance is below 80% | System highlights names and attendance percentage in red |  |  |
| 006 | Export student attendance percentage | September 2025  Student A attendance 90%  Student B  attendance 95% | The system gives a report (Excel/PDF) with each student’s name, total school days, days present, and attendance percentage |  |  |
| 007 | Check if admin can view student information | Admin logs in and selects a student record | The system shows the student’s details, such as name, course and other |  |  |

# **Software Configuration Management**

### **Repository Link:**https://github.com/ootri03/Software-engineering-assignment-

### **Description:**

Git helps by keeping a complete history of every changes you make to your files. You can save versions with commits, compare or revert to older ones if something goes wrong, and create branches to experiment without affecting the main project. This makes it easier to manage different versions of your work and collaborate with others safely.