   
**Mobile Device Programming**

**Mobile Application Report**

Report  
 4th December 2023

**Academic Year**: 2023-2024

**Name**: Soo Min Hao

**Student ID:** 20297525   
 **Module Convenor:** Dr Nabil El Ioini

**Introduction**  
  
 In 2020, we saw the big leap in the era of digitalization, the ability to organize and coordinate activities within an organization digitally is not just an advantage but a necessity. The Digital Arts Guild (DAG), a vibrant club under the Student Association of the University of Nottingham Malaysia, has been a production house for the creatives since its inception in 2010. As of 2023, it is home to five distinct departments—Video, Photo, Audio, Arts, and Marketing which boasts a plethora of talents and holds many specialized skills. However, with a diversity of operations comes a complex set of challenges that can and has impeded its efficiency and performance.

Managing such a multiple specialised department in an organization involves a labyrinth of tasks including event management, financial tracking, communication, and documentation. The club's current system, which relies on disjointed, decentralised platforms and manual coordination, has led to a consequential and disastrous disarray. Crucial information gets lost, deadlines are missed, and the professional image DAG strives to maintain is put at risk. These issues not only threaten the quality of work produced but also the club's reputation and, ultimately, its very existence.

Recognizing the urgency of the situation, the proposal for a dedicated mobile application is not just as a solution but as a strategic initiative to revitalize the club's operational dynamics. The envisioned application aims to address the core challenges faced by DAG: streamlining event preparations, enhancing communication, centralizing data storage, reducing dependency on limited resources, and reinforcing client trust. By providing a centralized, user-friendly platform tailored to the club's needs, the application seeks to elevate the standards of organization and professionalism within DAG.

The goal is to empower the members of DAG with a tool that not only simplifies the complex web of their daily operations but also fosters an environment where creativity and efficiency coexist harmoniously. By doing so, DAG can continue to thrive, sustain its esteemed reputation, and maintain its competitive edge in the vibrant landscape of university clubs.

**Project Planning and Management**

|  |  |  |
| --- | --- | --- |
| **Project Phase** | **Timeline** | **Description** |
| Phase 1: Pre-Development | 3rd October - 17th October 2023 | **Research and Requirements Gathering**  Conduct surveys and interviews with DAG members to gather detailed requirements.  Research similar applications for feature and design ideas.  Finalize project scope and application features. |
|  | 18th October 2023 | **Proposal Submission**  Submit initial proposal for the application development.  Await feedback from the Student Association and refine requirements as needed. |
| Phase 2: Design and Development Planning | 19th October - 1st November 2023 | **Design Workflows and Wireframes**  Create wireframes for each screen of the application.  Design user flow diagrams to map out the user journey within the app. |
|  | 2nd November - 12th November 2023 | **Development Planning**  Outline the development plan, including technology stack and development methodology.  Set up development environment and tools.  Create a detailed work breakdown structure and assign tasks to team members. |
| Phase 3: Core Development | 13th November 2023 | **App Designing Proposal**  Submit the app’s design proposal based on initial feedback before the 11.59pm deadline. |
|  | 14th November - 30th November 2023 | **Development Sprints**  Execute development sprints following Agile methodology.  Regularly test the application and review progress in sprint meetings.  Continuously integrate new features and test units. |
| Phase 4: Testing and Refinement | 1st December - 7th December 2023 | **User Testing and Feedback**  Deploy beta version of the application for select DAG members.  Collect feedback and conduct usability testing sessions.  Refine application based on user feedback. |
| Phase 5: Final Preparations | 8th December - 9th December 2023 | **Final Touches and Packaging**  Conduct final testing, including stress tests and security checks.  Polish UI/UX elements and ensure application stability.  Prepare the application package for submission. |
|  | 13th December 2023 | **Final Application Submission**  Submit the final version of the mobile application.  Ensure all documentation and necessary deployment details are included. |

**Requirements Analysis**

The development of the "DAGApp" involves a comprehensive set of requirements that cater to the needs of the Digital Arts Guild (DAG) club. These requirements are divided into functional and non-functional categories and are detailed through user stories and acceptance criteria.

**Functional Requirements**

1. User Authentication:

* Users must be able to register and log in to the app.
* The app should provide logout functionality.

1. Profile Management:

* Users should be able to view and edit their profile information (name, role, contact number, email).

1. Event Management:

* Users need the capability to view upcoming events and details.
* Authorized users should be able to create, edit, or delete event information.

1. Inventory Management:

* Users should have access to view items in the inventory.
* Authorized personnel must be able to add, update, or remove inventory items.

1. Reminder System:

* The app should display reminders for upcoming tasks or events.
* Users should be able to add and delete reminders.

1. Department Information:

* Users need to view information about different departments within the club.
* The app should facilitate direct communication (e.g., phone calls) by clicking on the contact details of department heads.

**Non-Functional Requirements**

1. Usability:

* The app should be user-friendly with an intuitive interface.
* Accessibility features for differently abled users.

1. Performance:

* The app should load content within an acceptable time frame.
* Efficient handling of data to minimize latency.

1. Security:

* User data should be encrypted and securely stored.
* The app should comply with data protection regulations.

1. Compatibility:

* The app should be compatible with most modern Android devices.
* It should support different screen sizes and orientations.

**User Stories**

1. As a club member, I want to log into my account, so that I can access personalized content.

**Acceptance Test: User can successfully log in using their credentials and view personalized content.**

1. As the event manager, I want to add new events to the app, so that members are informed about upcoming activities.

**Acceptance Test: The president adds an event, and it becomes visible to all members on the event page.**

1. As the club’s president, I want to view department contacts, so that I can easily communicate with department heads.

**Acceptance Test: Member clicks on a department, views the contact information, and initiates a call through the app.**

1. As the studio manager, I need to update the inventory list, so that the club’s assets are accurately tracked.

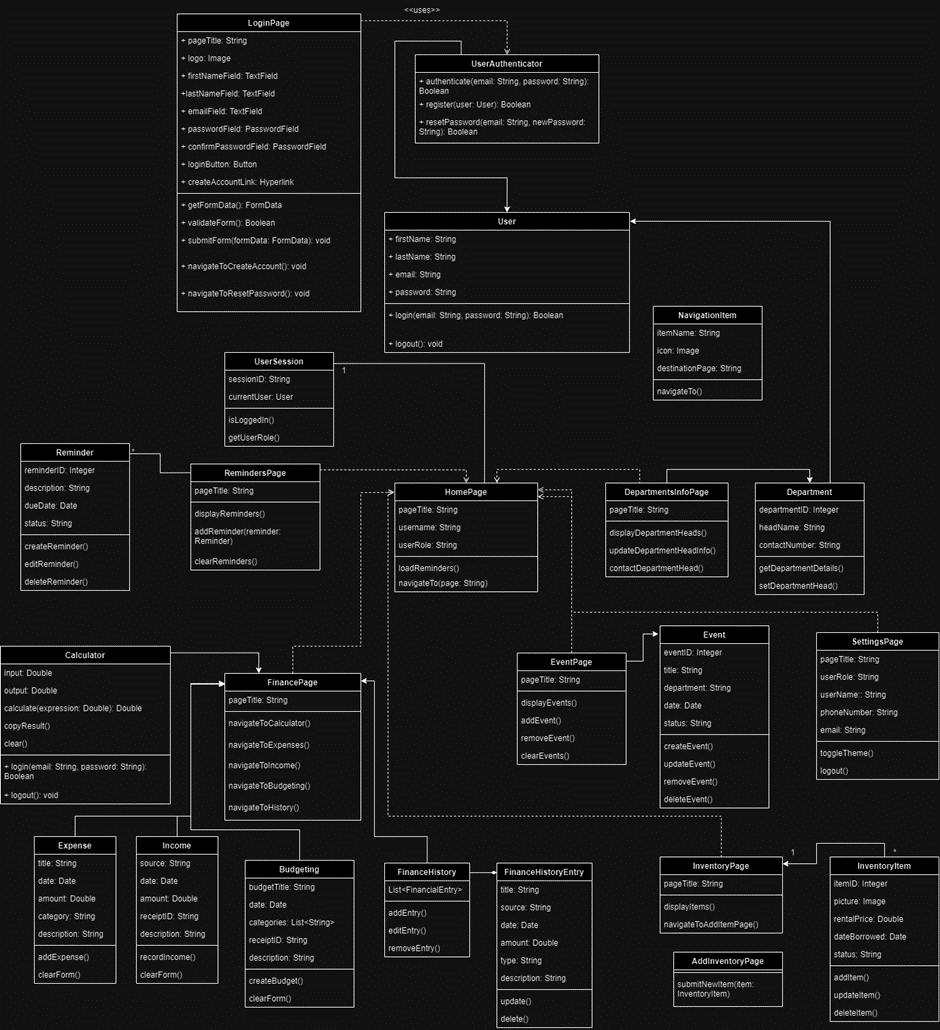
**Acceptance Test: Inventory manager adds a new item to the inventory, and it is reflected in the inventory list.**

1. As a user, I want to receive reminders, so that I do not forget important tasks and events.

**Acceptance Test: User adds a reminder, and it is displayed on the app’s home screen on the specified date and time.**

1. As an administrator, I need to edit user profiles, to ensure that member information is up to date.

**Acceptance Test: Administrator edits a user profile, and changes are immediately reflected in the app.**

**Design and Architecture**  
  


**Design Principles and their Rationale  
  
MainPage (Login Page)**A screenshot of a login screen

Description automatically generated

**Principle:** Clarity and Simplicity

**Rationale:** The login page adheres to clarity with a simple layout that focuses on essential elements. By providing fields for the first name, last name, email, and password, it clearly delineates what information is necessary for login, ensuring a user-friendly experience without overwhelming users with unnecessary details.

**Signup Page**  
A screen shot of a login form

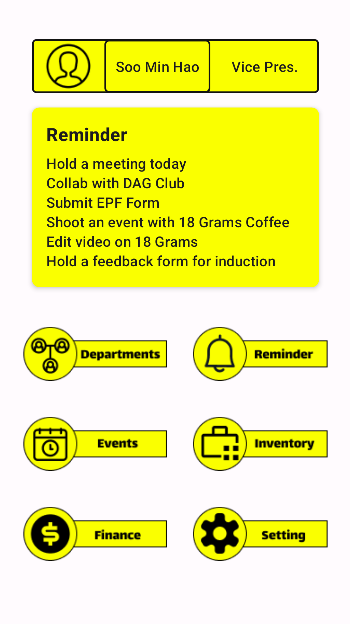
Description automatically generated  
 **Principle:** Consistency and Predictability

**Rationale:** The signup page maintains consistency with the login page, using the same colour scheme and input field styles. This design choice helps users predict the signup process based on their login experience, facilitating ease of use, and reducing the learning curve.

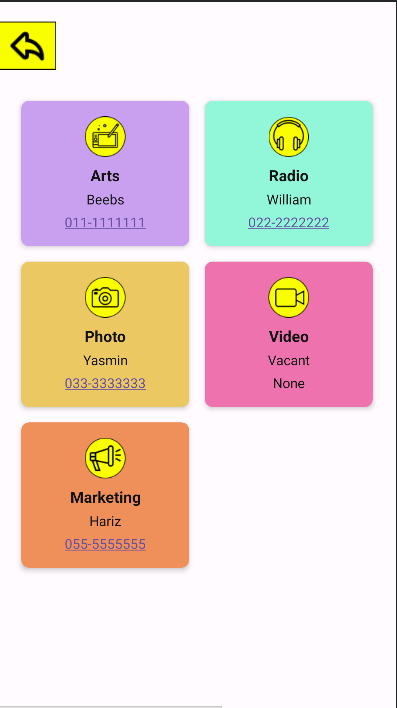
**Reset Password Page  
A screenshot of a login screen

Description automatically generated  
  
Principle:** User Control and Freedom

**Rationale:** The reset password page provides users with the control to recover their account access autonomously. It offers a straightforward approach to reset passwords, empowering users and enhancing their trust in the application's security.

**HomePage**  
  
  
**Principle:** Visibility of System Status and Recognition Rather Than Recall

**Rationale**: The homepage displays the user's name and role prominently, providing immediate feedback on who is logged in. It includes intuitive icons for main features like departments and reminders, enabling users to recognize and navigate the app with ease rather than remembering where each feature is located.

**DepartmentInfoPage**  


**Principle:** Aesthetic and Minimalist Design

**Rationale:** The Department Info Page uses a grid of colourful cards to represent different departments, adhering to an aesthetic design that enhances usability. The minimalist approach ensures that users are not distracted by extraneous information and can focus on the primary task of accessing department details.

**Events**  
A screenshot of a phone

Description automatically generated

**Principles:** Consistency: The events page uses a consistent layout for each department with the same sequence of "Event Title," "Event Date," and "Status" fields. This consistency allows users to quickly become familiar with the interface and know what to expect when interacting with different sections.

Contrast: Each department is represented with a distinct color, improving the visual distinction between different categories. This helps in quickly identifying the department sections and enhances the user experience by making navigation more intuitive.

Hierarchy: The use of size and boldness for the department titles serves to establish a visual hierarchy, making it clear that these are headers for the sections below. The 'Add' button is prominently placed at the bottom, signifying its importance as the primary action on the page.

Affordance: The text fields are clearly marked with borders and background colors that afford text entry. The 'Clear' and 'Add' buttons are designed to look clickable, which suggests their functionality.

Alignment: The alignment of the text fields and buttons creates a structured grid that is visually pleasing and easy to navigate. This grid layout aids in organizing the information in a manner that is easy for users to process.

**Rationale:**

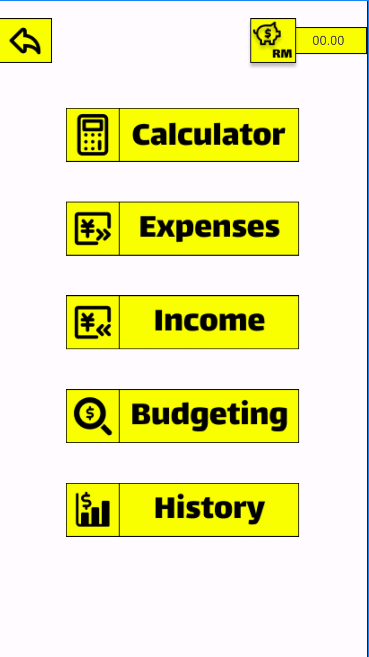
Colour Coding: Each department has a specific colour, which can help users to quickly locate and associate the events with their respective departments.

Return Navigation: The back arrow at the top left corner follows the common design pattern for navigation, allowing users to easily return to the previous screen.

Input Fields: The input fields are given ample space, which makes it easier for users to interact with the interface, especially on touch devices.

Action Buttons: The 'Add' and 'Clear' buttons at the bottom of the page are designed to indicate final actions that users can perform after entering data, following a logical flow of data entry followed by submission or cancellation.

Simplicity: The design avoids clutter and focuses on the task at hand, which is the entry and management of event information. This simplicity helps in reducing cognitive load and potential user errors.

**Finance Main Page**  


**Principle:** Flexibility and Efficiency of Use

**Rationale:** The Finance Main Page provides direct access to critical financial features such as expenses, income, budgeting, and history. This layout caters to both novice users and experienced users by allowing quick navigation, improving the efficiency of financial management tasks.

**Finance Calculator Page**   
**A screenshot of a computer

Description automatically generated**

**Principle:** Consistency and Feedback

**Rationale:** The calculator interface is straightforward, with clearly labelled operation buttons that provide immediate feedback when pressed. The use of bright yellow for operation buttons and input fields ensures that they are easily distinguishable from the result display. This design choice adheres to the principle of recognition rather than recall, allowing users to interact with the calculator intuitively.

**Finance Income Page**   
**A screenshot of a phone

Description automatically generated**

**Principle:** Aesthetic and Minimalist Design

**Rationale:** The Income interface avoids unnecessary information, focusing on the essential fields like Title, Date, Amount, Category, and Description. The use of contrasting colours for the input fields and background helps users to easily identify where to enter data. The 'Add' button is prominently displayed, guiding users towards the next step in the data entry process.

**Finance Expenses Page**   
**A screenshot of a phone

Description automatically generated**

**Principle:** Error Prevention

**Rationale:** Like the Income interface, the Expenses section uses clearly labelled fields to prevent errors during data entry. The color-coded fields ensure a minimalistic approach, reducing the potential for visual clutter that could lead to mistakes.

**Finance Budgeting Page**   
**A screenshot of a phone

Description automatically generated**

**Principle:** User Control and Freedom

**Rationale:** The Budgeting interface empowers users with clear options to add new budgeting data. It maintains consistency with the rest of the app’s design, offering a familiar experience that users can control. The 'Add' button is a direct call-to-action, which provides users with the freedom to add entries at will.

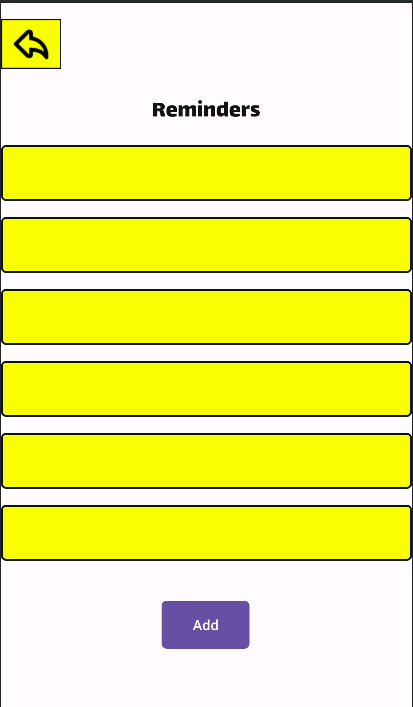
**Finance History Page**

A screenshot of a computer screen

Description automatically generated

**Principle:** Visibility of System Status

**Rationale:** The Finance History interface provides a summary view of the different financial categories: Income, Expenses, and Budgeting. Each section is color-coded for easy distinction and provides a snapshot of the most recent entries, keeping the user informed of their financial history's status immediately.

**Reminders Page**  


**Principle:** Error Prevention

**Rationale:** The Reminders Page is designed with a simple 'Add' button to minimize user errors when inputting reminders. The clear division of each reminder entry allows for easy review and editing, thus reducing the risk of overlooking or forgetting important tasks.

**Inventory**  
A group of colorful labels

Description automatically generated with medium confidence  
**Principles:**

Grouping/Similarity: The Inventory page groups items by equipment type, using similarity in colours and layouts to associate related items. This grouping follows Gestalt principles, making it easier for users to understand which items belong together and process the information as part of a related set.

Contrast and Colour Coding: Different equipment types are distinguished using distinct colour codes. This helps users quickly differentiate between categories, such as camera and audio equipment, enhancing the visual organization of data.

Hierarchy: The titles for each equipment type (e.g., "Camera Equipment") are prominently displayed, creating a visual hierarchy that guides the user's eye and indicates the start of a new section.

Affordance: The 'Update Inventory' button is clearly designed to suggest click-ability, using a contrasting colour that stands out against the background. This indicates that it is an actionable element.

Alignment: The inventory items are aligned in a grid layout, providing a structured and organized interface. This aligns with the principle of alignment, which states that items should be visually connected with shared edges or orientations.

Repetition: The repetition of input fields across different categories creates a rhythm that users can quickly get used to. It simplifies the form-filling process as users know what to expect with each new section.

**Rationale**

Usability: The layout is simple and intuitive, minimizing the learning curve for new users and enhancing the overall usability of the app.

Efficiency: By grouping related items together and using a consistent layout, the design allows for quick data entry and review, promoting efficient interaction.

Aesthetics: The use of colon not only has functional benefits but also makes the interface more visually engaging.

Flexibility: The design provides a flexible template that can be easily expanded or adapted for additional equipment categories if needed.

**Settings Page**  


**Principle:** User Control and Freedom

**Rationale:** The Settings Page gives users control over their profile and app settings, with the ability to change their details and toggle between light/dark mode. The inclusion of a prominent 'Logout' button respects user autonomy, providing a clear exit path from the user's account.

**App Implementation  
  
Account Activity**

**MainActivity.java (activity\_main.xml)**

The MainActivity serves as the entry point of the app. In the XML layout file activity\_main.xml, the user interface for the main screen is defined. It usually contains views such as buttons, text inputs, and other interactive elements that the user can interact with upon launching the app.

**ResetPasswordActivity.java (activity\_reset\_password.xml)**

The ResetPasswordActivity is likely responsible for allowing users to reset their passwords. This involves interacting with the activity\_reset\_password.xml layout which would contain form fields such as 'Old Password', 'New Password', and 'Confirm New Password', alongside a button to submit the request.

**SignUpActivity.java (activity\_sign\_up.xml)**

The SignUpActivity manages the user registration process. The activity\_sign\_up.xml layout file would define the user interface for the sign-up form, which typically includes fields like 'Username', 'Email', 'Password', 'Confirm Password', and a 'Sign Up' button to complete the registration process.  
  
**Home Activity**  
  
**HomeActivity.java (activity\_home.xml)**

Serves as a dashboard after login, it would present key information and offer navigation to other app features like DepartmentInfo, Events, Finances, Reminders, Inventory and Settings. The XML layout would have a user-friendly interface with access points to various functionalities.

**Department Activity**

**DepartmentInfoActivity.java (activity\_department\_info.xml)**

This component might display information about different departments within the organization or app context. It would include a list or grid of departments, possible using a GridView for layout. The corresponding Java file would manage the display of department data, handle user interactions, and navigate to details or editing screens.

**AddDepartmentInfoActivity.java (activity\_add\_department\_info.xml)**

This would handle the changes in heads of department information, with fields to input details such as departments’ head names and their contact info. The Java class would contain methods to validate and save this information to the app's data store.

**Finance Activities**

**FinanceMainActivity.java (activity\_finance\_main.xml)**

**CalculatorActivity.java (activity\_calculator.xml)**

**IncomeActivity.java (activity\_income.xml)**

**ExpensesActivity.java (activity\_expenses.xml)**

**BudgetingActivity (activity\_budgeting.xml)**

**FinanceHistoryActivity (activity\_finance\_history.xml)**

These activities are related to financial management within the app. FinanceMainActivity serves as a hub for financial features such as a calculator, income records, expense tracking, budgeting, and viewing recent finance history. Each activity would have its own XML layout, providing the necessary input fields and display formats required for financial calculations and record keeping.

**Event Management Activities**  
**ClubEventsActivity.java (activity\_club\_events.xml)**   
**RemindersActivity.java (activity\_reminders.xml)**

These activities are used for managing events and reminders in a club or group setting. The XML layout files define the user interface for creating, viewing, and managing these events and reminders.  
  
**Inventory Management Activities**

**ClubInventoryActivity.java (activity\_club\_inventory.xml)**

This component likely manages an inventory for a club or organization. This could include tracking items, rental prices, and status. The XML file would contain the layout for inventory management features.  
  
**Account Management and App Settings Activities**  
  
**SettingsActivity.java (activity\_settings.xml)**

Manages app settings that could include preferences for account management, save account info and logout. The XML layout file would provide the interface for these settings.

**Application Testing**

1. Unit Testing:

While typically automated, unit testing in manual contexts involves checking individual functions and methods for expected behaviour. Due to its granular nature, manual unit testing can be time-consuming and is often limited to critical business logic that requires thorough validation.

Results and Fixes:  
Due to time constraint, unit testing wasn’t implemented.

2. Functional Testing:

This involves testing each function of the application by providing input and verifying the output against the functional requirements.

Results and Fixes:

Tested all user flows, such as sign-up, sign-in, password reset, and navigation between different activities.

Identified a bug where the reset password flow was not correctly verifying user input, which was fixed by adjusting the validation logic.

3. Integration Testing:

Integration testing checks the data flow and control between modules of the application.

Results and Fixes:

Tested the integration between the FinanceMainActivity and other finance-related activities. Discovered issues with data not correctly passing between screens, which was resolved by debugging the Intent data transfer.

4. System Testing:

System testing looks at the application's behaviour, ensuring that all components work together as expected.

Results and Fixes:

Performed end-to-end testing scenarios that mimic user behaviour. Found discrepancies in the UI flow, which were corrected by reordering the activity transitions.

5. Regression Testing:

After any fix or feature addition, regression testing ensures that new code changes have not adversely affected existing functionality.

Results and Fixes:

After implementing fixes for data handling in financial activities, regression tests were conducted to ensure no new bugs were introduced. No additional issues were found.

6. User Interface (UI) Testing:

UI testing focuses on the visual and interactive aspects of the application.

Results and Fixes:

Checked for UI consistency across different activities, ensuring that elements like the bank\_balance EditText correctly reflect the updated data. Adjusted the XML layouts for uniform appearance.

Found issues with responsiveness on different screen sizes and fixed them by using scalable units for dimensions and testing with constraint layouts.

7. Usability Testing:

Involves real users interacting with the application to evaluate the ease of use, navigation, and overall user experience.

Results and Fixes:

Conducted with a small group of users from the target audience. Feedback on navigation complexity led to redesigning the navigation flow for better intuitiveness.

8. Performance Testing:

Assesses how the app performs under stress or heavy data loads.

Results and Fixes:

Tested app performance with multiple concurrent financial transactions. Discovered and fixed a slowdown due to inefficient database queries.

Through rigorous manual testing, several issues were identified across different aspects of the app. These were systematically addressed, and subsequent rounds of testing showed marked improvements. Ensuring a bug-free user experience was achieved by applying thorough test scenarios, reflecting real-world usage as closely as possible. This approach helped in delivering an application that met the functional and non-functional requirements, with a strong emphasis on user satisfaction and system robustness.

**User Documentation**

**Overview**

The mobile application, designed primarily for the Digital Arts Guild (DAG) at the University of Nottingham Malaysia, serves as a comprehensive tool for club management. It includes features for managing departments, events, finances, reminders, inventory, and user settings, all tailored to streamline club operations.

**Getting Started**

1. Installation: The app is available for Android devices and can be installed through the Google Play Store. Ensure your device meets the minimum system requirements.
2. Creating an Account: Upon launching the app for the first time, users are prompted to create an account. This involves entering personal details (name, email, etc.), creating a password. Selecting a role within the club (e.g., President, Treasurer) is available through settings.
3. Login: Returning users can log in using their registered email and password. A 'Forgot Password' feature is available for users who need to reset their credentials.

**Navigating the Home Page**

After logging in, users are directed to the Home Page, which provides a dashboard view of various functionalities:

* Profile Section: Displays the user's name and role. Users can tap their profile to view and edit their personal details.
* Reminders: A list of upcoming tasks and reminders specific to the user's role.
* Navigation Icons: Quick access icons to various features like Departments, Events, Finance, Inventory, and Settings.

**Feature Usage**

1. Managing Departments:

* Access contact information for department heads.
* Update leadership roles or add new contacts.

1. Organizing Events:

* Create and view upcoming events.
* Manage details such as event title, department, date, and status.
* Financial Management:

1. Use the Finance tab to monitor and record club expenses and income.

* Access a built-in calculator for budgeting and other monetary calculations.
* Review and update the club’s financial history.

1. Inventory Tracking:

* Maintain an updated list of club assets.
* Add new items, update the status of existing items, and manage rentals.

1. Setting Reminders:

* Set and manage reminders for tasks, meetings, and events.
* Personalize reminders based on date, time, and priority.

1. Customizing App Settings:

* Adjust personal settings such as contact information and display preferences (light/dark mode).
* Option to log out of the app for security.

**User Roles and Permissions**

The app is designed for use by club executives with distinct roles and permissions. These roles dictate the level of access and the functionalities available to each user, ensuring secure and role-specific interaction with the app’s features.

**Assumptions and Limitations**

* The app assumes a basic familiarity with smartphone usage and Android operating systems.
* Data is managed offline; users are responsible for manual data entry and editing.
* Real-time collaboration and internet-based features are not included in the initial release.

**Future Development**

Future updates may introduce cloud-based features, real-time updates, and additional functionalities based on user feedback and technological advancements.

**Steps of Application Deployment**

The deployment phase is a critical step in the lifecycle of the mobile application, as it involves making the app available to end-users. The deployment process for my mobile application includes several key steps, ensuring that the app is tested, secure, and ready for public availability.

**Pre-deployment Checklist**

Before deploying the app, I will ensure that the following pre-deployment checklist is completed:

**Code Review and Testing:**

All code has undergone thorough review and testing, including unit tests, integration tests, and user acceptance testing (UAT).

**Security Measures and Ethics:**

The application has been checked against common security vulnerabilities, and all user data handling complies with relevant data protection laws.

**Performance Optimization:**

The app will undergo extensive benchmarking in performance and any issues that requires necessary optimizations will be made to ensure smooth operation for users’ experience.

**Quality Assurance and User Testing**:

I will select a couple of users for final testing before launching and will ensure that the app has signed off as defect-free and meeting all functional requirements.

**Documentation**:

I will include all documentation, such as the user guide, technical documentation, and API references, and ensure they are updated and ready for release.

**Deployment**

**Environment Setup:**

Ensuring that production servers and databases are set up with the appropriate configuration alongside the backend services are deployed on scalable cloud services to handle expected user loads.

**App Store Submission:**

I will prepare the app store listing with compelling descriptions, screenshots, and marketing material before submitting the app to the respective app stores, such as Google Play Store for Android and Apple App Store for iOS, following their submission guidelines and requirements.

**Continuous Integration and Delivery (CI/CD):**

I will implement a CI/CD pipeline to automate the build, test, and deployment process by setting up automated scripts that can compile the code, run tests, and deploy the app to production servers.

**Monitoring:**

I will integrate monitoring tools to track the app's performance and usage statistics in real-time and set up alerts for any critical incidents that might affect the user experience.

**Backup and Recovery:**

I will ensure that backup systems are in place for the app's data and services and regularly test recovery procedures to confirm that the app can be quickly restored in case of a failure.

**Post-deployment**

After the app is deployed:

**User Feedback:**

I will continue to monitor user feedback closely for any issues that might have been missed during testing for couple of months before to ensure that initial release of app is executed well and having a list of users wanted features to be included in the future.

**Updates and Patches:**

I will be prepared to quickly deploy updates and patches in response to any issues faced by the app through user feedback or new requirements.

**Marketing and Promotion:**

I will coordinate with the marketing team to launch the app through various channels to maximize visibility and downloads.

**Maintenance**

The app will enter a maintenance phase where regular updates, bug fixes, and performance enhancements will be released as part of version management. The development team will support the operational team by providing necessary technical support and updates as required.

**Assumption**

**Initial Release of Application Assumption**  
**Offline and User Independence-** Users will not need to rely on an active internet connection to access and interact with the app's core features, enhancing usability in environments with poor or no connectivity.

**Data Entry and Editing-** All data entry and modifications will be performed manually by the user. Automated data sync or retrieval from an online server will not be available at initial release of app.

**Data Storage-** The application will store data locally on the user's device. Users are responsible for the backup and transfer of their data if needed.

**Functionality Limitation-** Features that typically require internet access, such as real-time collaboration, update, push notifications, or cloud backups, will be excluded from the initial version.

**User Base Assumptions**

**User Having Knowledge in Smartphone Usage:** It is assumed that users have basic knowledge of smartphone usage and familiarity with Android Operating Systems as well as with the inclusion of standard UI elements like buttons and swipe gestures. It is also assumed that the app is primarily used within the university setting by students and faculty who are members of the Digital Arts Guild.

**Operational Assumptions**

**User Roles and Permissions:** It is assumed that the app will be used by individuals with clearly defined roles (e.g., Treasurer, President, Event Manager) and that users will adhere to the permissions associated with their role.

**User Behaviour:** It is assumed that users will enter dates up to date for accurate record-keeping and that they will use the app's features as intended without attempting to circumvent any processes.

**Technical Assumptions**  
 It is assumed that users have Android devices capable of running the application smoothly, with a certain minimum Android OS version.