AIN SHAMSUNIVERSITY - FACULTY OF ENGINEERING
International Credit Hours Engineering Programs -iCHEP

MAJOR TASK

Hedieaty

CSE 431 - Mobile Programming – Fall 2024 – Project Specification Document V2.3

Hedieaty is a gift list management app designed to streamline the process of creating, managing, and sharing wish lists for special occasions such as birthdays, weddings, engagements, graduations, and holidays. The app features a user-friendly interface that allows users to easily add gifts to their lists, either manually or through an integrated barcode scanner. With Hedieaty, users can manage their lists flexibly, including adding, deleting, and modifying items at any time—except for items that have been pledged. The app also enables users to publish their lists to the cloud, allowing friends and family to view and pledge to purchase gifts, enhancing the joy of gift-giving. The app should be designed using Flutter and published on Appstore, Google Play Store, and/or Amazon App Store.

Requirements

#1. UI Layout and Design (Minimum 6 Pages) [25 marks] 27.5

- Home Page:

- A list of friends displayed with their names and profile pictures.
- For each friend, indicate if there are any upcoming events created (e.g., "Upcoming Events: 1" or "No Upcoming Events" or use numbers like whatsapp msg numbers).
- Clicking on a friend's name navigates to their gift lists, allowing users to view details of their lists and pledge gifts.
- ~~Users can add friends via phone numbers manually or from their contact list.~~
- Search functionality to find friends' gift lists.
- At the top of the home page, include a prominent button to create a new event or gift list, labeled "Create Your Own Event/List."

- Event List Page:

- Display list of events, sortable by name, category, and status. (Upcoming/Current/Past)
- Buttons for adding a new event, editing existing events, and deleting events.
- Editing Support: Users can modify/delete event details at any time.

- Gift List Page:

- Display of gifts associated with the selected event, sortable by name, category, and status.
- Buttons for adding new gifts, editing existing gifts, and deleting gifts.
- Visual indicators for pledged gifts (color-coded).
- Editing Support: Users can add, delete, or modify gifts until they are pledged.

- Gift Details Page:

- Input fields for gift name, description, category (e.g., electronics, books, etc.), and price.
- Option to upload an image of the gift.
- Status toggle (available, pledged) with a restriction that pledged gifts cannot be modified.

- Profile Page:

- User profile management with options to update personal information and notification settings.
- List of user's created events and associated gifts.
- Link to My Pledged Gifts Page

- Same Layout for user's Event List Page / Gift List Page / Gift Details Page can be used for Friend's pages with the additional feature of

- Option to pledge specific gifts from the friend's list.
- Users are Notified when friends pledge gifts to the user's list.

- My Pledged Gifts Page:

- Overview of gifts that the user has pledged, along with friend names and due dates for each pledged item.
- Options to modify the list if the pledge is still valid (pending gifts).

#2. Database Integration [10 marks] 11.5

- Implement SQLite for local storage of user events, gift lists, gift details, and user profiles.
- Structure tables for:
- Users (ID, name, email, preferences)
- Events (ID, name, date, location, description, user ID)
- Gifts (ID, name, description, category, price, status, event ID)
- Friends (User ID, Friend ID)

#3. Real-time Database Integration [15 marks] 16.5

- Use Firebase Realtime Database for syncing published gift lists across devices.
- Implement user authentication with Firebase Authentication to manage user accounts and secure data.
- Allow users to update the status of gifts (e.g., changing from "available" to "pledged" or "purchased") at real-time.
- Color-code gifts based on their status (e.g., green for pledged, red for purchased) updated at real-time.

#4. Barcode Scanner Integration [10 marks] 0

- Integrate a barcode scanning functionality using a Flutter plugin, enabling users to scan products in stores and automatically add them to their gift list.

#5. Quality Enhancement Features [15 marks] 16.5

- Introduce animations for transitions between pages and visual feedback for user actions
- Implement a notification system using Firebase Cloud Messaging to alert the gift list creator when another user pledges to buy a gift.
- In-app notifications for updates on gift status changes.
- Develop advanced querying features for searching and filtering gifts by name, category, or event date.
- Code structure follows clean architecture.
- Enable data validation to ensure the integrity of user inputs (e.g., valid date formats, required fields).

#6. Auto-Test Script [10 marks 11.5

- Create a test suite that contains a number of testcase using Flutter's testing framework, including unit tests and integration tests for app workflows. The list of test cases should be written in the project document.
- Create an auto-test script (Powershell test script that runs from the command prompt in or bash script in Linux that runs from linux shell) to automate the test procedure and to run the testcases and generate logs without human interaction through the utilization of adb commands
- Package the test scripts alongside the app for easy execution so that running the auto-test script can run all testcases and generate the test results in a log file.

#7. Documentation and QA [15 marks] 16.5

- Provide comprehensive project document detailing app functionality and user guides.

- The document should include the following sections:)

  - Introduction,
  - Survey of Similar Apps in the market with comparative analysis of the advantages and disadvantages of each app.,
  - UI Design including detailed description of and screenshots of the UI pages,
  - Code Description and Navigation scenarios
  - Test plan and scoreboard of testcases,
  - Known bugs,
  - Version Control Activity Report. On GitHub, navigate to the main page of the repository. There are two ways to enter the activity view: On the main page of the repository, to the right of the list of files, click Activity. Alternatively, click Branches, then to the right of any branch, click .
  - The app should be published on Appstore, Google Play Store, and/or Amazon App Store and a link to the store landing page should be added to the project document.

- Students should prepare 2-minute video, post it on youtube and demonstrate it to the instructor. Provide the link to the video in the project document.

- The project should be maintained using version control software (such as git).
  A Screenshot of the git activity and link to the online private repo should be added to the project document. Access to the instructor should be given to allow viewing of commit days/times and version control utilization.

Project Submission Instructions

Submit two incremental versions of the project PDF reports to LMS (or Teams if instructed to do so).

a**. The first milestone** would be the full layout of the app design (DART) – Requirement #1. You should copy the DART code in a Word file, add screenshots of the layout of all pages, and list the components used in the layout. Following that, convert the file into PDF and submit it on LMS. **The first milestone is not graded.**

b. **The second milestone** is the basic function of the application of navigation to all pages and contents on all pages along with database integration (Requirements #2, #3). The Dart code should be copied in a Word file, add screenshots of the layout, and list the testcase scenarios. Following that, convert the file into PDF and submit it on LMS. **The second milestone is not graded.**

c. **Final Delivery** A full functioning application with all requirements (1-7) as listed in the specification document.

Final Delivery submission

- Upload the final project report as a PDF
- Upload the project workspace folder to LMS.
- Upload the final Android .apk file. <rename the file student_name.apk>
- Upload test package (powershell test script that runs from the command prompt in windows or bash script in Linux that runs from linux shell + test environment that utilize adb commands)
- Create a 2-minute video demo of the application and post it on youtube. Provide the link in the project document.
- Demonstrate your app to your lab/course instructor

Marking Rubric

Component	Comments	Marks
Requirement #1	Evaluation is based on completeness of requirements, correct functionality, following best practices, clean commented code, User friendly interface, input validation, correct state management, handling screen orientation changes and different screen sizes.	**25** 27.5
Requirement #2	Evaluation is based on completeness of requirements, correct functionality, following best practices, clean commented code, correct structure of DB, efficient queries, and appropriate conflict strategy management.	**10** 11.5
Requirement #3	Evaluation is based on completeness of requirements, correct functionality, following best practices, clean commented code, correct implementation of authentication and DB rules, efficient sync of local and cloud DB, handling corner cases, such as offline usage.	**15** 16.5
Requirement #4	Evaluation is based on completeness of requirements, correct functionality, following best practices, clean commented code, correct integration of a barcode scanning functionality using a Flutter plugin, input validation.	**10** 0
Requirement #5	Evaluation is based on completeness of requirements, correct functionality, following best practices, clean commented code, clean architecture for app structure, correct integration of features.	**15** 16.5
Requirement #6	Evaluation is based on completeness of requirements, correct functionality, following best practices, clean commented code, correct packaging of test scripts and clear instructions for execution in a readme file.	**10** 11.5
Requirement #7	Evaluation is based on completeness of requirements in terms of project report documentation and Version control. A single commit to the repo or just few commits prior to the submission date doesn't fulfill this requirement. Activity log should reflect proper utilization of version control. Commits should ideally start enough time before the due date of the first milestone and continue to progress throughout the semester. Tags should be created for MS1, MS2, and Final Delivery.	**15** 16.5
Total		**100**

- Late submission of the final delivery (up to 3 days) results in 50%-mark deduction penalty.
- Late submission more than 3 days is not accepted and results in 0 marks for the project.
- Submissions might be checked against plagiarism and AI-generated content in the code or the project report. 0 marks will be granted in case of plagiarism or AI-generated content.