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

Students can choose one of the following options

MAJOR TASK – Option 1

BEACON 

Broadcast Emergency Alerts Community Offline Network

MAJOR TASK – Option 2

SMART Hydroponic 

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

MAJOR TASK – Option 1

BEACON

Broadcast Emergency Alerts Community Offline Network

CSE 431 - Mobile Programming – Fall 2025 – Project Specification Document V4.0

BEACON: Broadcast Emergency Alerts Community Offline Network is a revolutionary disaster response communication app that creates resilient communication using peer-2-peer communication technology when traditional infrastructure fails. During natural disasters, power outages, or network failures, BEACON automatically connects nearby smartphones to enable users to share critical information, coordinate resources, and request help without relying on cellular towers or internet connectivity. The app facilitates real-time emergency alerts, and chat services. With an intuitive emergency-optimized interface designed for high-stress situations, BEACON ensures that when disaster strikes and traditional communication channels go dark, users can still communicate, coordinate, and care for each other through the power of peer-to-peer connectivity. To test this app, students need at least 2 physical Android devices.

Requirements

#1. UI Layout – design guidelines [3 marks]

BEACON's user interface follows a three-page navigation flow designed for intuitive emergency communication. The landing page presents users with two primary actions: joining an existing emergency communication or initiating a new communication. Upon selection, users navigate to the network dashboard that displays all available/connected devices within the selected range. From this central hub, users can either send a pre-defined message to the connected device or initiate private conversations by selecting individual users, which launches the dedicated chat interface. Throughout all pages, voice command functionality enables hands-free operation - essential during emergency situations where visual attention may be limited. There is a fourth page for resource sharing coordination (medical supplies, shelter, food), and a fifth page for user profile and Emergency contact setup.

#2. Mobile Technology [2 marks]

The BEACON app uses peer-to-peer communication. This can be done through WiFi Direct, Nearby Share, or any other similar technology, with the app implementing automatic peer discovery within the technology range. A literature review of the used technology and analysis of the target communication protocol is an important objective in this requirement.

#3. Database Integration [3 marks]

- Implement SQLite for user profile, network activities, list of connected devices with timestamps of last seen active and any other useful data. Database should be encrypted when the app is inactive. State management should be done using Provider or Bloc.

#4. Quality Enhancement Features [2 marks]

- Support for Text to speech and speech recognition.
- Implement a notification system to alert the host when a new device joins the network.
- Code structure follows MVVM architecture.
- Enable data validation to ensure the integrity of user inputs (e.g., valid date formats, required fields).

#5. Auto-Test Script [3 marks]

- 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.

#6. Documentation and QA [2 marks]

- 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.,
  - Literature review on the implementation of the target technology (i.e. WiFi Direct, Nearby Share, etc.) in Mobile Apps supported with citations of at least 15 references.
  - 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 from GitHub.
  - Work statement that explains the contribution of each team member in the project.

- 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 folder (Zipped) 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 (unlisted). Provide the link in the project document.
• Demonstrate your app to your lab/course instructor on the scheduled demo day.

NOTES:

- All requirements MUST be submitted to LMS as per the submission instruction.
- The project MUST be demonstrated to the instructor on the scheduled project demo day.

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 for all 5 screens to support all features, handling screen orientation changes and different screen sizes.	**3**
Requirement #2	Evaluation is based on completeness of requirements, correct implementation of the target technology (WiFi Direct, Nearby Share , etc.),review of the technology specs and analysis of the target protocol.	**2**
Requirement #3	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. Also includes encryption, state management	**3**
Requirement #4	Evaluation is based on completeness of requirements, correct functionality, following best practices, clean commented code, correct integration of a notification, TTS and SR functionality, data validation and following MVVM design pattern.	**2**
Requirement #5	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 of regression test in a readme file. Includes unit, widget and e2e.	**3**
Requirement #6	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 will NOT fulfill this requirement. Auto generated Survey from AI models results in zero marks. 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.	**2**
Total		**15**

- **NOTES**
- 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.
- Failure to present your app to the instructor on the scheduled project demo day results in **0** marks in the project.
- Failure to answer the technical question raised during the presentation in a way that implies that student doesn't have enough knowledge about the delivered project results in **0** marks.
- Students have the option to work in groups up to 5 students after getting approval from the course instructor provided that the group will submit a statement of work that explains the tasks completed by each member in the group.

Student Name	
Student Number	
Course	**CSE 431 – Fall 2025**
Major Task	**Mobile Application using Flutter (BEACON)**

Evaluation Sheet

Requirement	Mark	Out of	Comments
UI Layout and Design 5 screens with all required features		**3**	
Using P2P technology correctly (WiFi Direct, Nearby Share , etc.), review of the technology specs and analysis of the target protocol		**2**	
SQLite local database, encryption, state management		**3**	
TTS & SR, Notification, MVVM, Data Validation		**2**	
Testing: auto-test script with documented regression test cases (unit, widget, e2e)		**3**	
Documentation and QA		**2**	
Total		**15**	
Course Instructor	**Dr. Haytham Azmi**		

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

MAJOR TASK – Option 2

SMART Hydroponic

CSE 431 - Mobile Programming – Fall 2025 – Project Specification Document V4.0

SMART Hydroponic is Flutter mobile application that monitors and controls a small-scale hydroponic growing system. The application will serve as a central dashboard for real-time sensor monitoring, automated system control, and data visualization. The system integrates hardware sensors with cloud-based data storage and processing using Firebase services.
The app will enable users to:

- Monitor critical environmental parameters in real-time
- Control hydroponic system components remotely
- View historical data trends and analytics
- Receive alerts and notifications for system anomalies
- Configure automated control thresholds and schedules

Requirements

#1. UI Layout – design guidelines [3 marks]

SMART Hydroponic UI design includes several key screens and features. The **Splash Screen & Authentication** section incorporates an app logo with loading animation, user login/registration, and password recovery. The **Dashboard** displays real-time sensor data, system status, critical control buttons, and the current mode (manual or automatic). The **Sensor Monitoring** screen shows individual sensor readings with status indicators, refresh/auto-update options, and calibration settings. The **Control Panel** allows manual actuator control, scheduling, emergency stops, and logs control history. The **Analytics & History** screen presents charts, historical data, export options, and trend analysis. The **Settings** screen manages sensor thresholds, notifications, system calibration, and user profiles. Finally, the **Alerts & Notifications** screen provides real-time alerts, severity filtering, notification history, and acknowledgment features.

#2. Mobile Technology [2 marks]

SMART Hydroponic is designed to interface seamlessly with hardware components through Firebase Realtime Database or Firestore for data communication and control. The hardware setup requires a microcontroller platform such as the Arduino Uno, ESP32, or ESP8266, equipped with WiFi connectivity, sufficient GPIO pins for sensor and actuator integration, and appropriate power regulation (5V/3.3V). The system must include at least three sensors from the following: temperature sensors (e.g., DS18B20 or DHT22) to monitor ambient and water temperature, water level sensors (e.g., ultrasonic or float type) for nutrient reservoir monitoring, pH sensors to track solution acidity, TDS/EC sensors to measure nutrient concentration, and light intensity sensors (e.g., LDR or TSL2561) for automated lighting control. Additionally, students must implement control for at least two actuators, such as a water pump for nutrient circulation, LED grow lights for plant lighting, cooling fans for temperature regulation, or nutrient dosing pumps for automated nutrient delivery. All sensor data and actuator commands must be transmitted and managed via Firebase, ensuring real-time synchronization between the app and the hydroponic system hardware. An alternative option is to use masterofthings or similar online dashboard for IoT simulation.

#3. Database Integration [3 marks]

- Implement SQLite for sensor and actuators activities, with timestamps of last warnings or alerts from the system and any other useful data, such as scheduled tasks, etc. State management should be done using provider or Bloc.

#4. Quality Enhancement Features [2 marks]

- Support for Text to speech and speech recognition.
- Implement a notification system to alert the host when a critical warning such as high temperature or low water level is received.
- Code structure follows MVVM architecture.
- Enable data validation to ensure the integrity of user inputs (e.g., valid date formats, required fields).

#5. Auto-Test Script [3 marks]

- 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.

#6. Documentation and QA [2 marks]

- 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 from GitHub.
  - Work statement that explains the contribution of each team member in the project.

- 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 folder (Zipped) 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 (unlisted). Provide the link in the project document.
• Demonstrate your app to your lab/course instructor on the scheduled demo day.

NOTES:

- All requirements MUST be submitted to LMS as per the submission instruction.
- The project MUST be demonstrated to the instructor on the scheduled project demo day.

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 for all screens to support all features, handling screen orientation changes and different screen sizes.	**3**
Requirement #2	Evaluation is based on completeness of requirements, correct communication with the hardware/masterofthings dashboard through firebase.	**2**
Requirement #3	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. Also includes state management	**3**
Requirement #4	Evaluation is based on completeness of requirements, correct functionality, following best practices, clean commented code, correct integration of a notification, TTS and SR functionality, data validation and following MVVM design pattern.	**2**
Requirement #5	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 of regression test in a readme file. Includes unit, widget and e2e.	**3**
Requirement #6	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 will NOT fulfill this requirement. Auto generated Survey from AI models results in zero marks. 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.	**2**
Total		**15**

- **NOTES**
- 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.
- Failure to present your app to the instructor on the scheduled project demo day results in **0** marks in the project.
- Failure to answer the technical question raised during the presentation in a way that implies that student doesn't have enough knowledge about the delivered project results in **0** marks.
- Students have the option to work in groups up to 5 students after getting approval from the course instructor provided that the group will submit a statement of work that explains the tasks completed by each member in the group.

Student Name	
Student Number	
Course	**CSE 431 – Fall 2025**
Major Task	**Mobile Application using Flutter (SMART Hydroponic)**

Evaluation Sheet

Requirement	Mark	Out of	Comments
UI Layout and Design all screens with all required features		**3**	
Using firebase to communicate effectively with the dash/hw board		**2**	
SQLite local database, state management		**3**	
TTS & SR, Notification, MVVM, Data Validation		**2**	
Testing: auto-test script with documented regression test cases (unit, widget, e2e)		**3**	
Documentation and QA		**2**	
Total		**15**	
Course Instructor	**Dr. Haytham Azmi**		