SOFTWARE PROJECT PLAN

1.0 Introduction

This section provides an overview of the software engineering project, including its scope, major functions, performance goals, and management or technical constraints.

1.1 Project scope

The proposed project aims to build an iOS app that uses machine learning to inspect battery pouch cells and detect any defects or irregularities in their images.

The system will include three main parts:

- Input: Users can take photos or upload images of pouch cells using their iPhone camera or gallery.
- Processing: A trained convolutional neural network (CNN) model that processes each image, performs feature extraction, and classifies it as "defective" or "non-defective."
- Output: The app will display the result on screen, showing the classification, a confidence score, and a short explanation.

The application will provide real-time feedback and allow users to store, export, and review results. The project's objective is to combine machine learning and mobile app development to create a portable, intelligent inspection tool that supports research and industrial needs.

1.2 Major software functions

The system will include several main functions that cover the full development process, from collecting data to delivering the final app:

1. Data Collection and Annotation:
    a. Gather images of pouch cells
    b. Label each image by marking defects with bounding boxes and assigning classification tags.
2. Model Development
    a. Train and fine-tune a convolutional neural network (CNN) to detect defects.
    b. Test and optimize the model so it runs efficiently on mobile devices.
3. App Development:
    a. Create an iOS app using Swift and SwiftUI.
    b. Integrate the trained ML model into the app using CoreML for on-device predictions.
4. Testing and Feedback
    a. Perform both functional and performance testing to ensure accuracy and stability.
    b. Test the app with real samples and gather user feedback for improvements.
5. Final Integration and Reporting
    a. Complete all documentation, including a user guide and final report.

b. Deliver the final working prototype and presentation.

1.3 Performance/Behavior issues

The system is expected to meet the following performance goals:

- The machine learning model should achieve at least 90% accuracy on validation data.
- It should run smoothly on iPhone 13 or newer devices without noticeable lag.
- The app should process and classify each image in under 2 seconds to provide real-time feedback.
- The system should be able to store and manage at least 100 results without slowing down.

1.4 Management and technical constraints

The project will face several key constraints that affect development and delivery:

1. Time Constraint: The entire project must be completed and submitted by April 8, 2026.
2. Technical Constraint: Integrating the trained ML model into iOS using CoreML may require additional format conversion and optimization, which could restrict model size or complexity.
3. Dependency Constraint: Final app integration and testing depend on the completion and validation of the ML model, meaning this step must be done before full system testing.

2.0 Project Estimates

This section provides cost, effort and time estimates for the projects

2.1 Historical data used for estimates

Finding a source for historical data on a project like this is difficult. Sources I found involved a Quora post as well as a Reddit comment, both in general app development. The Quora post was about how long it takes to develop an iOS app. According to the post, depending on the complexity of the app, the average time-frame is 12-24 weeks. The reddit comment also seems to be around this estimate. The comment was made asking a question on a android dev subreddit, asking how long it takes from start to finish to create an app. The comment replying says it takes about 3-4 months (but that includes using APIs the commenter is familiar with). Searching for sources on Machine Learning development and training, an article was found stating this can take up 4-8 weeks, and integrating into an app would take 3-5 weeks.

As for monetary costs, assuming the team already owns the equipment needed, there should be no cost. Most, if not all tools used are free or have free versions, so no money will need to be spent on the project.

2.2 Estimation techniques applied and results

2.2.1 Estimation technique *Bottom-Up*

Stage	Estimated Effort (Person-Hours)	Duration (Weeks)	Deliverable
Data Collection & Annotation	80	5	Expanded and labeled image dataset
Model Development	100	7	Trained and optimized ML model
App Development	120	6	Working and robust iPhone app
Testing & Feedback	80	7	Validation report and feedback results
Final Integration & Reporting	60	6	Documentation and final presentation
Total	~440-460	~31 weeks	-

2.2.2 Estimate technique *Analogy-Based*

Based on historical data presented in 2.1, mobile app development projects require 12-24 weeks, while machine learning model training and integration takes about 7-13 weeks. Combining these ranges results in an expected total project duration for 25-32 weeks, which aligns closely with the planned project timeline of about 28 weeks.

2.3 Reconciled Estimate

After comparing both techniques, the reconciled estimate is as follows:

- Total Effort: ~ 450 person-hours
- Duration: ~ 31 weeks
- Team composition: 4 members (ML engineer, iOS developer, data annotator, tester)
- Project Cost: No cost

2.4 Project Resources

2.4.1 People

The team working on this project is a team of four. Each member has their own overall roles and responsibilities. One team member is developing the machine learning model, one is the data specialist, one is the front-end developer and one is documenting and testing. Everyone either has the previously mentioned skills or will be learning them on the fly. Although we have our individual roles, most of us are open to help others on their parts.

2.4.2 Hardware Requirements

- Macbook laptops are required for each team member.
- iPhone for testing (iPhone 13 or later).

2.4.3 Software Requirements

- Xcode + SwiftUI for front-end development.
- Python + Jupiter/Colab for AI model training and testing
- GitHub for version control.
- Trello / Notation for task tracking and progress updates.
- Google Drive for documents and data files.
- LabelImg / Roboflow / CVAT for image annotation.

2.4.4 Other Resources:

- Pre-labeled dataset of pouch cells.
- Access to Wi-Fi and storage.

3.0 Risk Management

This section discusses project risks and the approach to managing them.

3.1 Project Risks

Each project risk is described. The CTC format may be used.

3.2 Risk Table

The complete risk table is presented. Name of risk, probability, impact and RM3 pointer are provided.

3.3 Overview of Risk Mitigation, Monitoring, Management

An overview of RM3 is provided here. The Complete RM3 is provided as a separate document or as a set of Risk Information Sheets.

4.0 Project Schedule

This section presents an overview of project tasks and the output of a project scheduling tool.

4.1 Project task set

The project follows a waterfall development model. There are 5 main phases:

- Data Collection & Annotation
- Model Development
- App Development
- Testing & Feedback
- Final Integration & Report

4.2 Functional decomposition

- Data Collection & Annotation:
    - Define image capture protocol
    - Dataset Preparation
        - Annotate dataset using bounding box and classification tags
        - Perform QA of annotation and finalize train/validation/testing splits

- Model Development:
    - Model training:
        - Implement image preprocessing
        - Train baseline convolution neural network
        - Perform hyperparameter tuning

    - Model Evaluation and Deployment:
        - Evaluate performance
        - Export optimized model to CoreML format

- iOS Application Development:
    - Image Capture and Preprocessing:
        - Set up iOS skeleton in Swift/SwiftUI
        - Implement camera capture module
        - Add image preprocessing pipeline in app

    - Inference and Result Display:
        - Integrate CoreML model into the iOS app
        - Build inference pipeline
        - Design results display screen

    - Reporting Features:
        - Implement result history storage
        - Enable export/report feature

- Testing & Validation:
    - Functional and Performance Testing:
        - Create unit tests for preprocessing pipeline
        - Conduct functional testing of inference accuracy
        - Perform performance benchmarking

        - Field and User Testing:
            - Pilot test app with lab/industrial samples
            - Collect user feedback on usability and accuracy

- Final Integration & Reporting:
    - Documentation:
        - Write technical documentation
        - Write user guide for app usage
        - Prepare validation report

    - Presentation and Delivery:
        - Prepare final presentation slides and demo video
        - Conduct final team rehearsal
        - Deliver final submission

4.3 Task network

- Data collection must be completed before model training
- Model training must be completed before full app integration
- App development can be started before model training is complete, but full app integration requires model training to be complete.
- Testing precedes final delivery.

Data Collection → Model training

Model training → Full app integration

App development + Model training → Full app integration

Full app integration → Testing

Testing → Final delivery

4.4 Timeline chart

Stage of Development	Stage Completion Date	Deliverable	Deliverable Completion Date
Data Collection & Annotation	10/24/25	Expanded and labeled image dataset	10/24/25
Model Development	12/12/25	Trained and optimized ML model	12/12/25
App Development	1/9/26	Working prototype Robust and accurate iPhone app	12/17/25 1/9/26
Testing & Feedback	2/27/26	Validation report based on field testing results	2/27/26
Final Integration & Report	4/8/26	Technical documentation	4/8/26

5.0 Staff Organization

The manner in which staff are organized and the mechanisms for reporting are noted.

5.1 Team structure

Eman Sedqi- Team Lead, ML Developer

Nesreen Ismail- Data Specialist

Oquba Khan- Front-end Developer

Firas Abueida- Documentation & Testing Lead

5.2 Management reporting and communication

Our team uses Zoom and WhatsApp for communication. WhatsApp is used for more brief messages and quick updates, while Zoom is used for sharing our progress in detail. We meet on Zoom every Monday as a team.

We also meet in person after lecture when time is available.

6.0 Tracking and Control Mechanisms

This section outlines how the project will stay on track, meet its goals, and maintain high quality throughout its development.

6.1 Quality assurance and control

To make sure all project deliverables meet the required standards, a Software Quality Assurance (SQA) process will be followed. This process includes several key activities:

- Code Reviews: Weekly peer reviews on GitHub to check code quality, readability, and compliance with Swift and Python best practices.
- Testing:
    - Unit Testing: Tests individual components of the ML model and iOS app to ensure each function works as intended.
    - Integration Testing: Verifies that the CoreML model and iOS app work together correctly.
    - System Testing: Evaluates the entire app to confirm proper functionality and good performance.
    - Validation Testing: Uses real pouch cell images to confirm the model's accuracy and reliability in real-world conditions.

- Documentation Reviews: Ongoing reviews of all documentation to ensure clarity, accuracy, and completeness.
- Performance Benchmarks: Establish clear performance metrics, such as accuracy, processing speed, and storage efficiency, and also verify that the final product meets or exceeds these targets.

6.2 Change management and control

All project changes, whether they involve technology, functionality, or scheduling, will be managed through a structured Software Configuration Management (SCM) process to maintain consistency and control.

- Version control: GitHub will be used as the main platform for managing source code. All commits will be linked to assigned tasks in Trello to ensure traceability and accountability.
- Change request process:
    - A team member identifies and proposes a change.
    - The proposed change is discussed during a team meeting and documented.
    - The project lead reviews the change to assess its impact on the project's scope, cost, and schedule.
    - If approved, the change is implemented in a separate branch and merged into the main codebase only after testing and verification.
- Baseline Control:
    - Major deliverables, such as the dataset, trained model, and app code, will have controlled baselines. Any modification to these components requires prior approval before being integrated.

- Continuous monitoring:
    - Weekly progress reviews will be conducted to make sure that changes are properly managed and do not negatively affect the project's overall timeline or workflow.

7.0 Appendix

This section includes supporting information and reference materials relevant to the software project.

7.1 References

- Quora (2024). How long does it take to develop an iOS app? Retrieved from Quora.com (Reply from Maria Uidelli).
- Reddit (2024). How long does it take to build an Android app from start to finish? Retrieved from Reddit.com/r/androiddev.
- Debut Infotech (2025). Machine Learning App Projects: Timeline and Cost Factors. Retrieved from debuitinfotech.com.
- Apple Developer Documentation (2025). CoreML Integration Guide for SwiftUI Apps.

7.2 Tools and Technologies

Category	Tools/Technologies
Front-End	Swift, SwiftUI, Xcode
Machine Learning	Python, TensorFlow
Version Control	Git, GitHub
Collaboration	Trello, Notion, Google Drive
Dataset tools	LabelImg, Roboflow, CVAT
Testing	XCTest, PyTest
Reporting	Microsoft Word, Google Docs, Canva

7.3 Acronyms and Abbreviations

Acronym	Definition
ML	Machine Learning
CNN	Convolutional Neural Network
SQA	Software Quality Assurance
SQM	Software Configuration Management
UI/UX	User Interface / User Experience

Document Revision History

Version	Date	Description	Author
1.0	10/13/35	Initial draft of sections 1.0, 6.0, 7.0	Nesreen Ismail
1.0	10/13/35	Initial draft of sections 2.0, 4.0	Oquba Khan
1.0	10/13/35	Initial draft of sections 3.0, 4.0	Firas Abueida
1.1	10/14/35	Finalized for submission	Team