Software Requirements Specification

for

Machine Learning For Network-Denied Environments

Version 1.0 approved

Prepared by Sanjog Baniya, Jonathon M Dooley, Enrico Efendi, Wilson Gan,
Xavier Lara, Kevin Maravillas, Howard Nguyen,
Nisapat Poolkwan, Johnson Tan, Justin To, Alvin Yu

USC Institute for Creative Technologies / National Security Innovation Network

12/5/2023

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Justin To	12/5/2023	Initial draft of document	1.0

1. Introduction

The Software Requirements Specifications(SRS) provides a detailed outline of the software application designed for US Military seeking a solution for offline AI tutoring and offline content sharing or exchange. The document is necessary for guidance in the development process by describing functional and non-functional requirements of the software. This document emphasizes on what the software is aiming to attain, whereas its technical aspects of how these requirements will be implemented will be specified in the Software Design Document (SDD).

1.1 Purpose

The purpose of this SRS document is to outline the software requirements for a cutting edge AI powered tutoring application with offline capabilities. The software aims to address the problem faced in the areas where internet access is limited or unavailable, as defined in the first version of this release. The document shall cover all aspects of the software, laying a solid foundation for development, deployment and future updates.

1.2 Intended Audience and Reading Suggestions

The intended audience of this document are Software Developers, Liaisons, and Users.

- Software Developers: To understand the depth technical requirements that need to be met for the software implementation
- Liaisons: To assess the software's alignment with organizational needs and objectives
- Users: To help understand basic information pertaining to the software

To maximize the utility of this document, readers are advised to focus on sections pertinent to their specific roles.

1.3 Product Scope

This Image Classification software is an advanced mobile application designed to facilitate offline AI tutoring and content sharing for an organizational context. Its key features are:

- Offline AI Tutoring: Enables interactive and personalized learning experiences without access to the internet
- Content Sharing: Allows users to share resources within the app's ecosystem
- Automated Model Updates: Allowing automatic model update of the app to enhance accuracy when connected to the internet
- Data Collection for Enhanced Learning: It collects unlabeled data in offline mode for continuous improvement of the AI model

This software shall address the challenge faced in areas with limited or no internet access. It aims to provide a robust, user-friendly platform for effective learning, and knowledge sharing.

1.4 Definitions, Acronyms, and Abbreviations

- Refer to Appendix: A

1.5 References

- Tensorflow: https://www.tensorflow.org/tutorials/images/transfer_learning
- Keras: https://www.tensorflow.org/js/tutorials/conversion/import_keras
- Digit recognition in sklearn: https://scikit-learn.org/stable/auto_examples/classification/plot_digits_classification.html
- React Native: https://github.com/thecodingmachine/react-native-boilerplate
- Flask: https://flask.palletsprojects.com/en/2.3.x/quickstart/
- Python: https://www.python.org/

2. Overall Description

2.1 System Analysis

2.1.1 Project Goals

The proposed software application addresses the specific needs of the US Military, providing a comprehensive solution for offline AI tutoring and content sharing. The primary goals of the project are:

1. **Offline AI Tutoring:** Develop an advanced mobile application that facilitates interactive and personalized learning experiences without requiring internet access.
2. **Content Sharing:** Create a platform that enables users to seamlessly share educational resources within the application's ecosystem, fostering collaborative learning.
3. **Automated Model Updates:** Implement a mechanism for automatic model updates when the application is connected to the internet, ensuring the latest and most accurate AI capabilities.
4. **Data Collection for Enhanced Learning:** Enable the collection of unlabeled data in offline mode to continuously improve the AI model's accuracy and effectiveness.

2.1.2 Problem Statement

The software addresses challenges faced in areas with limited or no internet access, catering to the unique requirements of the US Military. The need for offline capabilities stems from the necessity to operate in environments where internet connectivity may be restricted or unavailable. The software aims to overcome this limitation by providing a robust and user-friendly platform for effective learning and knowledge sharing.

2.1.3 Technical Hurdles

The major technical hurdles associated with the project include:

1. **Offline Functionality:** Designing and implementing a seamless offline mode for AI tutoring, ensuring that the application remains fully functional without internet access.
2. **Data Synchronization:** Developing a reliable mechanism for synchronizing data, including model updates and user-generated content, between devices when an internet connection is established.
3. **Model Accuracy:** Ensuring the accuracy and efficiency of the AI model in offline mode, considering potential limitations in computing resources.
4. **Security:** Implementing robust security measures to protect sensitive data during offline and online interactions, particularly in military contexts where data confidentiality is paramount.

2.1.4 Solutions

To address these technical hurdles, the following solutions are proposed:

1. **Offline Functionality:** Implement a robust caching system that allows the application to store and retrieve necessary data locally, ensuring uninterrupted functionality without internet access.
2. **Data Synchronization:** Develop a secure and efficient synchronization protocol that allows seamless transfer of data between devices when an internet connection is available, prioritizing user-generated content and model updates.
3. **Model Accuracy:** Optimize the AI model for resource efficiency, and provide a mechanism for automatic updates when connected to the internet to enhance accuracy over time.
4. **Security:** Employ encryption protocols and authentication mechanisms to secure data during transmission and storage, adhering to military-grade security standards.

This system analysis provides a high-level overview of the project's goals, the challenges it addresses, and the proposed solutions to overcome technical hurdles. The subsequent Software Requirements Specification (SRS) document will detail specific requirements to guide the development process.

2.2 Product Perspective

Softwares that may have some similarities are any that involve identifying an object in a picture such as an application like Google Lens. Differences than other applications would be not needing an internet connection to identify the object in the picture. The motivation for creating this software is to enable the use of an application that can identify an object with AI/machine learning in an environment without an internet connection.

2.3 Product Functions

Web Client:

1. **Data Flow:**
    - The web client will use the server's web API to retrieve, update, and perform actions on the server.
    - Data to and from the server will flow through designated API endpoints.
2. **Caching and Display:**
    - Retrieved data will be stored in memory for modification and reading.

    - Modified data will be sent to the server, and cached data may be cleared if necessary.
3. **New Training Data Validation:**
    - Receive unverified data and information (images and labels) for the image classifier.
    - Display unverified image data in an image gallery.
    - Allow the user to edit and change labels based on a given list from the server.
    - Send confirmed unverified data, along with modified or unchanged labels, back to the server.
    - Signal the action to train the image classifier with newly staged data.
4. **Adding New Labels:**
    - Allow users to add new labels for the image classifier.
    - Enable labeling of images with no appropriate labels.
5. **Model Information:**
    - Provide information related to the model, including version, training set data, and updates.

Server:

6. **Receive Data:**
    - Receive data in the form of images from connecting devices.
7. **Incremental Model Training:**
    - Incrementally train the model using received images and their labels.
8. **Adjust Training Data:**
    - Possess the capability to adjust the data used for the training set.
9. **Security:**
    - May possess security measures to prevent the leak of information.

Mobile Client:

10. **Image Selection:**
    - Allow the user to select an image from the mobile device's photo gallery or take an image from the mobile device's camera.
11. **Image Processing and Classification:**
    - Process the selected image for classification.
    - Classify the selected image using a pre-trained machine learning model.
12. **Display Predictions:**
    - Capture and display the top predictions for the classified image.
13. **Error Handling:**
    - Handle errors related to image selection and classification.
14. **User Interface:**
    - Display the selected image in the user interface.
15. **Presentation of Results:**

    - Present the classification results in a user-friendly format.
16. **Communication with Server:**
    - Send the classified image data and predictions to the server.
17. **Nearby Communication:**
    - Should be able to communicate with other clients nearby for the purpose of sharing information.

This set of product functions outlines the major capabilities of the web client, server, and mobile client without delving into the specific implementation details. Each function is described in a way that is understandable to customers or stakeholders, providing a high-level overview of the software's capabilities.

2.4 User Classes and Characteristics

User classes include all general users who have a phone and install this app.

2.5 Operating Environment

The software will operate on all cell phones, including Android and IOS systems. It is supported on all platforms of the smartphone itself.

2.6 Design and Implementation Constraints

Tensorflow.js - a library for machine learning in Javascript. Tensorflow.js enables the development of ML models in JavaScript, and the use of ML directly in the browser or in Node.js.

Keras - an open-source library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library.

MobileNet - a type of convolutional neural network designed for mobile and embedded vision applications

React Native - open-source UI software framework. Used to develop Android and IOS applications

Flask - micro web framework written in Python

2.7 User Documentation

There are no current user manuals for the project.

2.8 Assumptions and Dependencies

- Users are expected to have a phone with a camera and a gallery application
- Users are expected to have an internet connection to download the application.

2.9 Apportioning of Requirements

- Implement a selected model for classifying objects.
- More training data for identifying different objects.
- A platform like an IOS/Android App.

3. External Interface Requirements

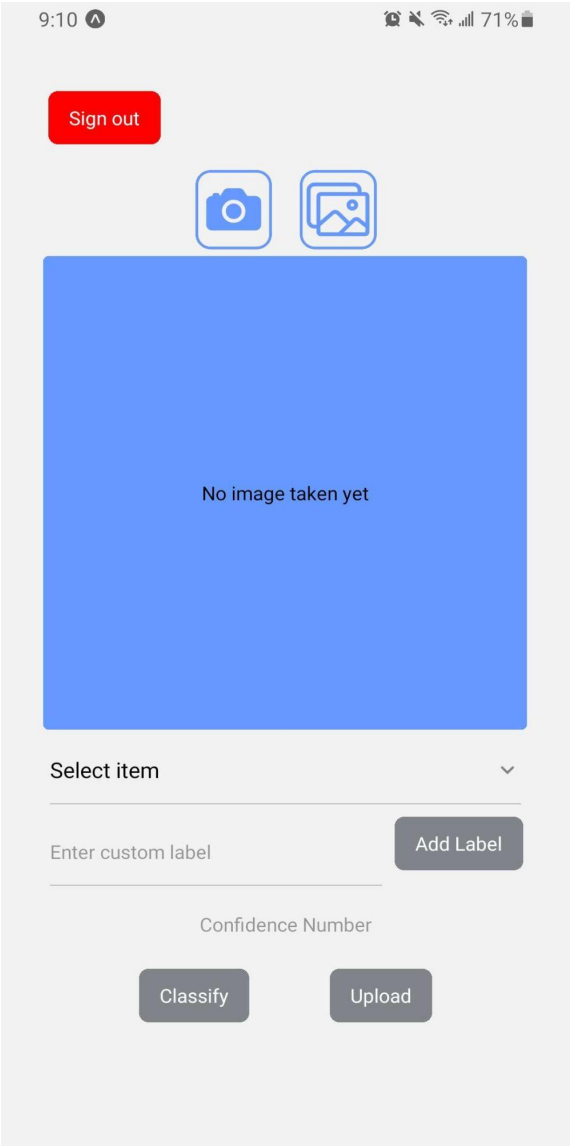
SERVER: N/A

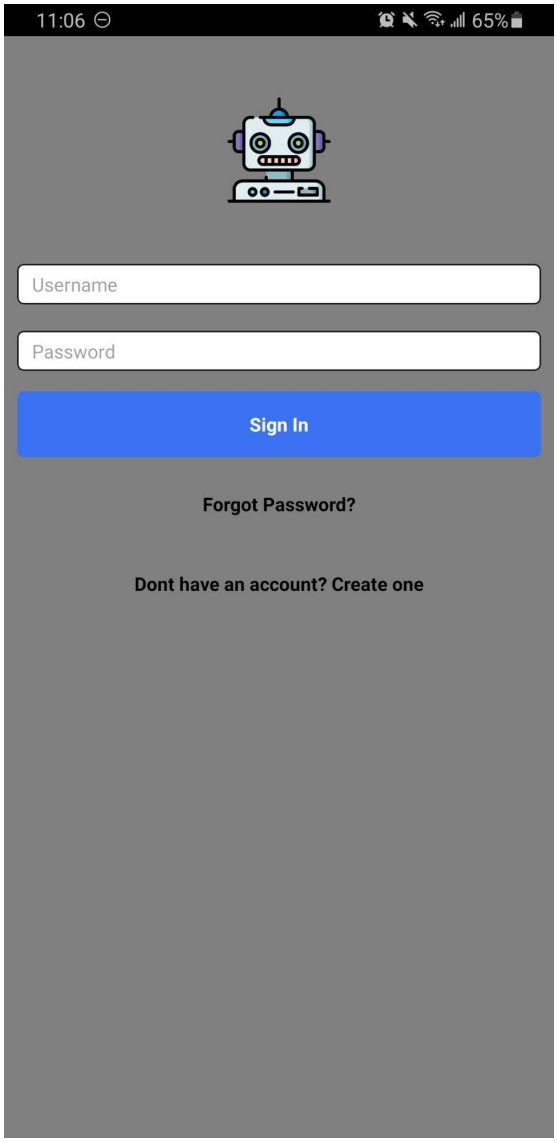
WEB: The web-client's main purpose is to interface with the server acting as the main admin to server actions and details. Thus the web-client's sole external interface will be with the server's API to signal actions or receive and update information and data.

MOBILE: The mobile client will get a trained model from the server using the endpoints provided by the server. With these endpoints, the mobile will be able to upload, download, and train data on a device. Using tensorflow.js, the mobile will be able to utilize the model that got downloaded to the device. This specific model will be a h5 file, in which the trained data will be stored.

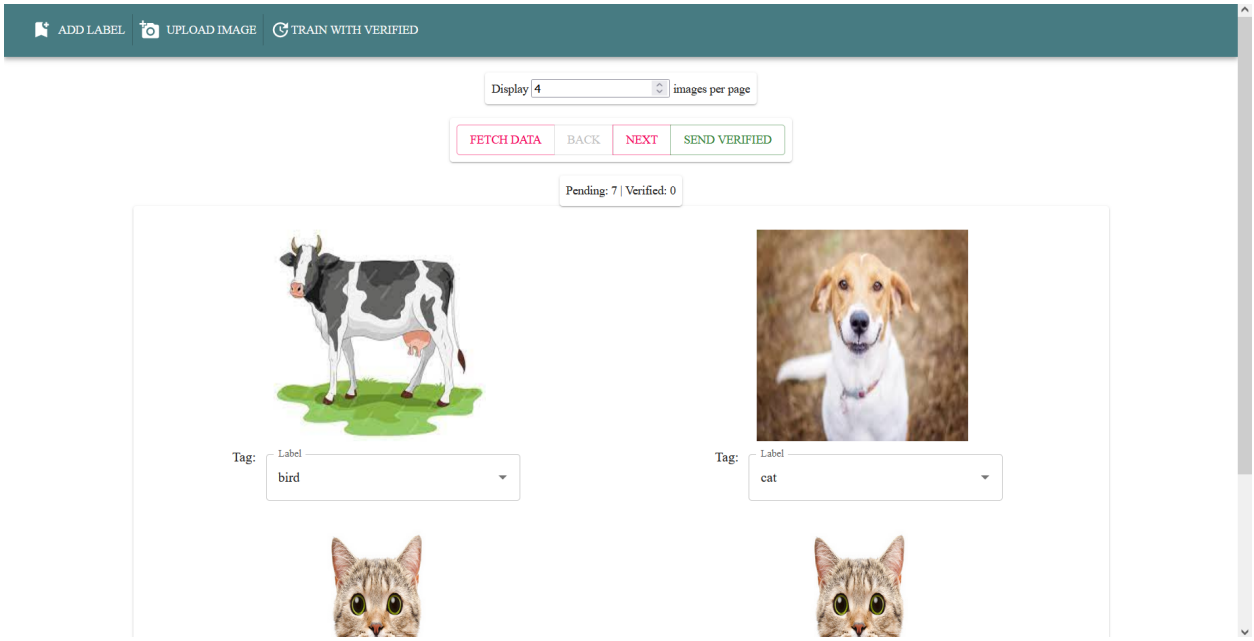
3.1 User Interfaces

Mobile Client:


9:10
71%
Sign out
No image taken yet
Select item
Enter custom label
Add Label
Confidence Number
Classify
Upload


11:06
65%
Username
Password
Sign In
Forgot Password?
Dont have an account? Create one

Web-Client:



3.2 Hardware Interfaces

Mobile Client: The logical characteristics would be the Image Picker Library that comes with Expo. The image picker library is compatible with both iOS and Android devices. The library facilitates the selection and retrieval of images from the device's local storage or camera. It returns the image data or file paths for further processing. The software interacts with the image picker library through function calls to trigger image selection and camera access. The library uses native APIs provided by the mobile operating systems to communicate with the device's storage and camera functionalities. The physical aspect would include the use of a camera. The app interacts with the camera hardware available on both iOS and Android.

3.3 Software Interfaces

Mobile Client:

- The main software uses React Native Ver. 0.72.5 and Expo Ver. 49.0.11 with other libraries including Expo ImagePicker

Web-Client:

- React with NextJS framework
    - This framework will be used for the frontend display of the web-client, and user interaction.

- Material UI
    - Main React package to stylize and provide basic modern web interfaces.
- Internal Server API
    - The primary software interface for which the web-client will utilize to communicate with the server.

3.4 Communications Interfaces

Mobile Client: During the registration, the user has to provide their e-mail along with other necessary information. As far as the app's main functions, we used the “formData” function to communicate directly to the server with image uploading. This transfers any information added to the object

Web-Client: The client will use basic http protocol to communicate requests to the main server. A REST style api will be provided as the main interface for data requests, updates, and triggering server actions.

4. Requirements Specification

4.1 Functional Requirements

Web Client:

4.1.1 : Data Flow - Server Web API

4.1.1.1 : The web-client shall utilize the server's web api as the main interface for data retrieval, updating, and actions performed on the server.

4.1.1.2 : Actions and function calls on the server will also be triggered through the API as well.

4.1.2 : Caching and Display

4.1.2.1 : The client, once retrieved requested data, shall store it in memory for modification and reading, in this case for displaying to the user.

4.1.2.2 : Once changes are made to the data, the client shall send it to the server and clear any data on memory if necessary.

4.1.3 : New Training Data Validation

4.1.3.1 : The web-client shall receive data and information (i.e. images and labels) of unverified and trained datasets for the image classifier.

4.1.3.2 : The web-client shall display unverified image data in an image gallery and allow the user to edit and change the labels on them based on a given list of labels listed by the server.

4.1.3.3 : The web-client shall send back any unverified data that is confirmed by the user to be verified alongside their changed or unchanged labels.

4.1.3.4 : The web-client shall signal the action to train the image classifier with any new data staged for that training.

4.1.4 : Adding New Labels:

4.1.4.1 : The web-client shall allow the user to add new labels for the image classifier to identify.

4.1.4.2 : The web-client shall also allow the user to label images when there were no appropriate labels for it.

4.1.5 : Model Information:

4.1.5.1 : The web-client shall provide any information related to the model, such as version, training set data, and updates to the model.

Server:

4.1.6 : The system shall receive data in the form of images from connecting devices

4.1.7 : Model

4.1.7.1 : The system shall incrementally train the model using the received images and their labels

4.1.7.2 : The system should possess the capability of adjusting the data used for the training set

4.1.8 : The system may possess security preventing the leak of information

Mobile Client

4.1.9 : The system shall allow the user to select an image from the mobile device's photo gallery or take an image from the mobile device's camera.

4.1.10 : The system shall process the selected image for classification.

4.1.11 : The system shall classify the selected image using a pre-trained machine learning model.

4.1.12 : The system shall capture and display the top predictions for the classified image.

4.1.13 : The system shall handle errors related to image selection and classification.

4.1.14 : The system shall display the selected image in the user interface.

4.1.15 : The system shall present the classification results in a user-friendly format.

4.1.16 : The system shall send the classified image data and predictions to the server.

4.1.17 : The system should be able to communicate with other clients nearby for the purpose of sharing information

4.2 External Interface Requirements

Web Client:

- **Server Web API:**
    - **Description:** The main interface with the server will be through a RESTful api architecture. All GET and POST requests should return or send data relevant to the request with some methods that signal an action on the server, such as to train with a data set.
    - **Source of Input/Output:** Receive from server, send from client
    - **Data Format:** JSON/String

Server:

- N/A

Mobile Client

- Image Selection Interface
    - Description of Purpose: Allows users to choose or take an image for classification.
    - Source of Input: User interaction with the mobile device's photo gallery or camera.
    - Valid Range: Any valid image format, primarily JPEGs and/or PNGs.
    - Accuracy/Tolerance: The system should handle common image sizes and formats.
    - Timing: On-demand when the user initiates image selection.
    - Relationships to Other Inputs/Outputs: Connected to the Image Processing module.
- Classification Results Interface
    - Description of Purpose: Display classification results to the user.
    - Destination of Output: Mobile device screen.
    - Valid Range: Predicted class names and confidence scores.
    - Accuracy/Tolerance: Displayed results should accurately represent the classification.
    - Timing: After the image classification process is complete.
    - Relationships to other inputs/outputs: Connected to Image Classification module.

4.3 Logical Database Requirements

This project will not utilize a database.

4.4 Design Constraints

Web Client

- N/A

Server

- Model limitations
    - Storage: The model used for classification must not require a large file size to store.
    - Update Capability: The model must be incrementally updatable.
    - Mobile Deployment: Lightweight enough to deploy on mobile devices.
    - Classification Accuracy: Should possess a sufficiently high prediction accuracy.

Mobile Client

- Hardware Limitations

    - Mobile Device: The application must operate within the constraints of mobile devices, considering limitations in processing power, memory, and storage.
    - Camera: The image classification process relies on the capabilities of the mobile device's camera, such as image resolution and quality.
- Operating System Limitations:
    - The application design must be compatible with iOS and Android. This includes utilizing platform-specific development frameworks and guidelines.
- Network Limitations:
    - Internet Connectivity: The application requires a reliable internet connection primarily for communication with the server for data transmission.
    - Data Usage: Potential constraints on data usage for users with limited data plans.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- Since this software requires the support of the mobile phone camera, users must use a mobile phone with a camera (if there is no camera, some functions will not be available)
- At the same time, the user's mobile phone will be requested to use WIFI, Bluetooth, roaming and other functions. If any are missing, some functions will not be available.
- The software only processes one image at a time. When a user uploads an image to our server, the mobile phone needs to have the requirement to log in to the network or use Bluetooth to transfer data to other user’s devices.
- When using the offline function to transmit data, the user needs to turn on the Bluetooth function.
- The software supports the login and use of multiple accounts, and the account of the software also supports the login of multiple devices.

5.2 Safety Requirements

For users, they need to understand that the images they upload will be stored in a database on our servers. Therefore, all users should be aware of the inherent risks of a data breach, which means their data will be exposed and exploited by others.

Therefore, all users should be aware of and fully understand the risks and consequences of sharing content on the Internet.

5.3 Security Requirements

The software requires users to register an account and log in to the account to use the functions of the software. Since users may need to upload images, users need to be careful to protect their account information from being stolen.

5.4 Software Quality Attributes

The software is always adapting since there are always new images that are being added to the server by the user itself. The software is portable as it is made easily available for both the computer and for mobile devices. The software is reliable as the software will send the new data that was accumulated to the server and will update the server once there is a stable network connection. The usability of the software is high as the desktop and mobile versions are easy to use for the general users as the interfaces are easy to navigate.

5.5 Business Rules

Since the software is image classification, there are no specific individuals that can perform specific functions. As of now, the software is meant to be easy to use so there are no individual-specific functions.

6. Legal and Ethical Considerations

This project is committed to upholding legal and ethical standards. Below mentioned considerations guide the development process, ensuring that the application is compliant with legal requirements.

- Data Privacy and Protection: Application collects and processes data like images and interactions. With adherence to data protection laws such as General Data Protection Regulation (GDPR), the application implements stringent data protection measures, ensuring transparency in data usage.
- Intellectual Property Rights: Application uses machine learning models and may incorporate open-source or third party software. Being compliant with copyright laws and open source licenses, it fosters a fair and creative environment.
- AI Ethics and Bias: Application uses AI, raises concerns about algorithmic bias and ethical implications of AI decision making. Regular auditing of AI algorithms for bias and fairness, and ensuring diverse and inclusive data sets for AI training, align with the ethical commitment to fairness and non-discrimination.

Appendix A: Glossary

Artificial Intelligence (A.I.): A technology aimed at getting machines to perform functions much in the same way as humans.

Machine Learning: A field of study branching off of A.I. that utilizes data analysis to learn without explicit programming.

TensorFlow: An open source framework developed by Google researchers to run Machine Learning, deep learning and other statistical and predictive analytics workloads.

Keras: An open source library that provides a python interface for artificial neural networks. Acts as an interface for the Tensorflow library.

Flask: A web framework; a python module to develop applications easily.

React Native: An open source UI software framework created by Meta, used to develop applications for Android, iOS, Web, Windows by enabling developers to use the React framework along with native platform capabilities.

Intellectual Property: A set of products protected under laws associated with copyright, patent, trademark, industrial design, and trade secrets.

GDPR: Stands for General Data Protection Legislation, governs the way in which we can use, process, and store personal data.

Open Source: Software where the creator allows other users direct access to the source code so the user can alter and distribute the software for their own purpose.

Interface: The place at which independent systems meet and act on or communicate with each other.