**Individual Software Project**

**Specification**

Mobile app for object detection in a museum

Airat Akhmadullin

Charles University, MFF

1. **Introduction**

This project aims to develop a cross-platform mobile application capable of processing images taken by users in a museum setting to detect and identify objects within the museum. The application will recognize exhibits based on a database of pre-recorded videos and photos. The primary objective is to achieve high accuracy and speed in object detection while considering the inherent limitations of mobile devices, including limited computing power and memory.

This application will enhance the museum experience by allowing visitors to take photos of exhibits and receive immediate, detailed information about them, thus enriching their knowledge and engagement.

1. **Analysis of Existing Programs**

Several existing applications provide museum guide solutions based on image recognition technology. Some examples include:

1. **Google Lens**:  
     
   Google Lens is an image recognition technology developed by Google, designed to bring up relevant information related to objects it identifies using visual analysis based on a neural network.

* **Strengths**: Uses advanced image recognition technology to identify objects and provide information from the web.
* **Weaknesses**: Requires a constant internet connection, introduces latency due to cloud-based processing, and is not specifically designed for museum environments.

1. **Smartify**:

Smartify is a mobile app which can scan artworks and return information about them.

* **Strengths**: Allows users to scan artworks in various museums to get detailed information, also provides audio information.
* **Weaknesses**: Relies heavily on internet connectivity for data retrieval and can be used only with paintings.

1. **Intended Functionality**

The application will allow users to take a photo of a museum object, identify the object by matching it against a database of images and videos, and retrieve detailed information about the object from a database. The app will operate offline to ensure accessibility and will feature a user-friendly interface optimized for fast and accurate recognition on mobile devices. Additionally, it is designed for Android with potential for future expansion to iOS.

1. **Main Functions**

**1. Photo capture and preprocessing:**

* Capture images using the device camera.
* Preprocess images (resizing, normalization) for efficient analysis.

**2. Feature extraction and matching:**

* Extract features from the captured image.
* Compare extracted features with those in the database to find the best match.

**3. Video retrieval and analysis:**

* Retrieve the corresponding video based on the matched features.
* Extract the object’s ID based on the retrieved video.

**4. Information display:**

* Query the database using the extracted ID.
* Display detailed information about the object to the user.

1. **Program Structure**

**Modules and Libraries:**

1. **Image processing module:**

* *OpenCV*: For image preprocessing and feature extraction (Python).
* *TensorFlow Lite*: For running the trained model on the mobile device (Kotlin).

1. **Database module:**

* *SQLite*: For local storage of object information.
* *JSON Parser*: For reading object information from JSON files (Python and Kotlin).

1. **User Interface module:**

* *Android UI Components*: For building the app interface (Kotlin).
* *Camera API*: For capturing photos (Kotlin).

1. **Machine Learning module:**

* *TensorFlow/Keras*: For training the object recognition model (Python).
* *TensorFlow Lite*: For running the trained model on the mobile device (Kotlin).

**Mutual Dependencies:**

* The Image Processing Module relies on *OpenCV* and *TensorFlow Lite*.
* The Database Module interacts with the local *SQLite database* and the *JSON parser*.
* The User Interface Module depends on *Android UI components* and the *Camera API*.
* The Machine Learning Module requires *TensorFlow/Keras* for model training and *TensorFlow Lite* for model inference on the device.

1. **OS, Language, and Development Environment**

*Operating System*: Android

*Programming Languages*:

* Python (for model training and data preparation)
* Kotlin (for Android app development)

*Development Environment*:

* Visual Studio Code
* IntelliJ IDEA

1. **Approximate Timeline**

* 22 June: baseline model is built in Python
* 27 June: accuracy is improved, final model is ready in Python
* 1 July: Kotlin application (without model) is built
* 3 July: model is integrated into application
* 5 July: application is tested
* 10 July: final improvements are done; application is ready for use

1. **Conclusion**

This specification outlines the detailed plan for developing a mobile application that enhances the museum experience by allowing visitors to identify and learn about exhibits through photos. By utilizing advanced image recognition techniques and ensuring efficient performance on mobile devices, this app will offer a unique and valuable tool for museumgoers.