Business Idea Summary: Global Photo Booth (GPB) Concept

Objective

To introduce a globally accessible photo booth experience, similar to the '인생네컷' (Life Four Cuts) concept, allowing users worldwide to easily take and print photos with customizable frames and features.

Core Concepts

QR Code Integration for Personal Devices

- **Mobile Access**: Users can scan a QR code at any participating location, connecting them to the store's server. This enables them to download or print their photos directly to their mobile devices.
- **Collaborations with Local Icons**: Partnering with local famous personalities or events (e.g., school football stars) to create custom frames or themes. This approach attracts more customers and provides a unique, localized experience that resonates with the community.

In-Store Photo Booth Installation (Hardware/Software)

- **Custom Booth Development**: Creating both the hardware (photo booth setup) and the software to operate it. The booth will feature:
    - High-quality cameras
    - Touch screens
    - Customization options for frames, filters, and other settings
- **Growth and Expansion**:
    - Scaling the business by installing booths in various locations.
    - Continuously improving software to meet customer needs, including:
        - Social media sharing options
        - Video recording capabilities

        - Mobile app integration for seamless photo access

Market Opportunity

The global popularity of photo booths for events, parties, and social media experiences creates a strong potential for growth. By offering an innovative and mobile-integrated solution, this concept can appeal to a wide audience, from casual users to those seeking personalized, localized photo experiences.

Key Benefits

- Provides a fun, interactive way for people to capture memories and share them instantly.
- Versatile product for a variety of venues, including:
    - Malls
    - Schools
    - Tourist attractions
    - More

Summary

This business idea leverages technology and creativity to create a unique, customizable photo booth experience that bridges global trends with local charm. Its scalability and adaptability make it a promising venture for diverse markets.

Service #1: Global Photo Booth (GPB)

System Architecture

The system is proposed as a simplified ***client-server architecture***. Mobile and web clients will serve as the primary means for user interaction. A Python-based backend will handle requests, process images, and manage data efficiently. Cloud integration will ensure scalable and minimal storage for data availability. Networking features, including QR code integration, will enable seamless connections between mobile clients and the backend.

- **Frontend:** Mobile and web clients for user interaction.
- **Backend:** Python-based server for handling requests, processing images, and managing data.
- **Cloud Integration:** Minimal cloud-based storage for scalability and data availability.
- **Networking:** QR code integration for linking mobile clients to the backend.

Programming Languages

For the implementation, the following languages have been chosen.

- **Backend: Python**: Efficient for handling a moderate load with libraries for image processing, API development, and database interaction.
- **Frontend Mobile: Kotlin Multiplatform Mobile (KMM):** For developing cross-platform mobile apps (iOS and Android) with a single codebase.
- **Frontend Web: Web Client:** HTML5, CSS3, JavaScript (or TypeScript) for basic web functionalities.

---

3. Frameworks and Libraries

The project will incorporate the following frameworks and libraries to streamline development.

- **Backend (Python):**
    - **FastAPI:** This framework will be used for building lightweight and responsive RESTful APIs.
    - **Pillow:** Image editing and processing will be facilitated through this library.
    - **OpenCV:** Advanced image manipulation may be implemented using this library, if required.
    - **SQLAlchemy:** Database management tasks will be handled using this ORM.
    - **qrcode:** QR codes for user-photo linking will be generated using this library.
    - **Celery (Optional):** Asynchronous tasks such as image processing may be managed using this tool.
- **Frontend (Kotlin for Mobile):**
    - **Kotlin Multiplatform Mobile (KMM):** This will enable a unified codebase for both iOS and Android applications.
    - **Jetpack Compose (Android):** The Android UI will be designed using this toolkit.
    - **SwiftUI (via KMM):** The iOS UI will be constructed using SwiftUI as part of KMM integration.
    - **Ktor Client:** API calls from the mobile application will be managed through this library.
- **Frontend (Web Client):**
    - React.js or simple HTML/CSS will be used for a lightweight web client if included.
- **Backend (Python):**
    - **FastAPI:** Lightweight framework for building RESTful APIs.
    - **Pillow:** For basic image editing and processing.
    - **OpenCV:** For more advanced image manipulation (optional).
    - **SQLAlchemy:** ORM for managing the database.
    - **qrcode:** For generating QR codes for user-photo linking.

        - **Celery (Optional):** For handling asynchronous tasks like image processing if needed.
- **Frontend (Kotlin for Mobile):**
    - **Kotlin Multiplatform Mobile (KMM):** Unified codebase for iOS and Android apps.
    - **Jetpack Compose (Android):** For building the Android UI.
    - **SwiftUI (via KMM):** For building the iOS UI.
    - **Ktor Client:** For handling API calls from the mobile app.
- **Frontend (Web Client):**
    - **React.js (or Simple HTML/CSS):** For building a lightweight web client if needed.

---

4. Database

The system will rely on the following database technologies.

- **Relational Database (for metadata):**
    - SQLite (lightweight, suitable for small-scale use cases).
    - PostgreSQL (if scalability is anticipated).
- **Cloud Storage (for photos):**
    - Amazon S3 or Google Cloud Storage (minimal configuration and cost-effective).

---

5. Cloud and Hosting

The cloud hosting of the system will be configured as follows.

- **Cloud Provider:**
    - AWS, Google Cloud, or DigitalOcean for hosting the backend and storage.
- **Server Hosting:**
    - Use a lightweight hosting service like Heroku or a VPS on

DigitalOcean for minimal load.

---

6. Security

The system’s security is prioritized with the following measures.

- **User Data Protection:**
    - HTTPS for secure communication between clients and server.
    - JWT (JSON Web Tokens) for user authentication.
    - AES Encryption for sensitive photo data during storage.
- **Compliance:** Security measures will adhere to global data protection regulations, ensuring user privacy.

---

7. Networking and QR Code Workflow

The QR code workflow is structured as follows.

1. **QR Code Generation:** The server will generate a QR code that links to the user's photo session on the backend.
2. **Mobile Access:** Users will scan the QR code to connect their mobile applications to the respective session.
3. **Photo Access:** Processed photos will be downloaded or shared via the mobile application.

---

8. Implementation Steps

The following steps outline the implementation process:

1. **Backend Development:**
   RESTful APIs will be built for user session management, image upload and

processing, and QR code validation. Image processing features will be integrated using libraries such as Pillow or OpenCV.

2. **Mobile App Development:**
   A mobile application will be developed with core functionalities, including QR code scanning, session linking, and photo customization. Kotlin Multiplatform Mobile (KMM) will be utilized for compatibility with both iOS and Android devices.
3. **Web Client Development:**
   If required, a basic web client will be developed for users preferring browser access.
4. **Cloud Deployment:**
   The backend will be deployed using Docker containers on cloud services such as AWS or Google Cloud.
5. **Testing:**
   The system will be rigorously tested with a limited number of users to validate its scalability, security, and usability.

This revised technical plan employs a clear structure with a focus on leveraging Python and Kotlin. The approach ensures a simplified, cost-effective solution suitable for the anticipated scale of the project.

This revised approach simplifies the system while meeting the requirements and leveraging Python and Kotlin effectively. It minimizes unnecessary complexity and focuses on scalable, cost-efficient solutions.