Product Requirements Document – *Photo Memories Enhancement App*

Overview:
A cross-platform **photo enhancement application** that helps **nostalgia seekers** and **family archivists** rediscover and rejuvenate old memories. Users can seamlessly **connect their photo libraries** (Google Photos, OneDrive, etc.) to search for past moments by text, person, or date, then apply advanced enhancements like **colorizing black-and-white photos**, **improving old scans**, and – as a standout feature – **animating still photos into short video clips**.[1] The app runs on **Web, iOS, and Android**, with an Azure-based C# backend and Blob Storage. A **freemium model** offers basic usage (1–2 enhancements free), with paid options for unlimited or premium features.

User Personas & Key Scenarios

- **Nostalgia Seeker:** An adult user looking to relive treasured moments (e.g. a parent recalling a 1990s vacation). They use the app to **find a specific memory** (by event name or date) and then **enhance it** – for example, colorizing a black-and-white childhood photo and turning it into a short animated clip to share with family. This persona values **ease of discovery** and quick transformations that make memories shareable.
- **Family Archivist:** A user who manages boxes of old family photos and videos. They scan vintage photos into the app and use batch **restoration features** to fix exposure, remove scratches, and upscale resolution. A key scenario is selecting a series of related photos (e.g. a wedding album) and using the app to **create a montage video clip** with gentle zooms and transitions. They also appreciate the ability to **tag people** (family members' faces) so they can later search by name and compile all media of a particular person over decades.

Core Features & Enhancement Capabilities

Smart Search & Browse: The app provides powerful ways to locate content:

- *Text Prompt Search:* Users can enter queries like *"Christmas 1998"* or *"beach trip college"* to find matching photos/videos. The app leverages captions, dates, and possibly AI image recognition to fetch relevant items.

- *People & Tags:* Through face recognition or metadata, users can filter media by person (e.g. see all photos of “Grandma Alice”). This makes it easy to gather memories of a specific individual.
- *Date Navigation:* A timeline or calendar view allows browsing by year or month (e.g. jump to **July 1975** to see photos from that period). This is especially useful for users digitizing chronological archives.

Photo Enhancement Toolkit: Built-in tools help modernize and improve old images:

- *Colorize Black & White:* Convert monochrome photographs into natural-looking color images automatically. The user simply selects an old B&W photo and applies *“Colorize”*; the app returns a full-color version.
- *Restore Quality:* Enhance faded or low-resolution images. This includes sharpening details, upscaling small scans (using super-resolution techniques), and removing artifacts (dust, scratches) from scanned prints. With one tap, an old photo becomes clearer and higher-definition.
- *Lighting and Corrections:* Basic editing adjustments (brightness, contrast, fade correction) might be auto-applied especially for aged photos that have yellowed or washed out over time. The goal is **one-click improvement** that yields a noticeably better image while preserving authenticity.

★ **Photo-to-Clip Animation:** *This is a signature feature of the product.* The app can **turn one or more photos into a short video clip**, adding motion and transitions:

- *Single Photo Animation:* Using AI, the app can animate a still photo – for example, creating a slow zoom and pan (the “Ken Burns effect”), or a subtle 3D parallax motion that brings depth to the image. In cases of portraits, it might even simulate minor movements (like a slight smile or blink) to make the subject appear alive. This feature **brings a static memory to life**, and is done with a simple “Animate” command. The user can preview a 5-10 second clip generated from a single image.
- *Multi-Photo Montage:* Users can select a set of related photos (e.g. five pictures from a birthday) and the app will automatically generate a dynamic slideshow video. It stitches the images together with transitions (cross-fade, pan from one to the next) and optional background music. This provides an **instant nostalgia reel** without any video editing skills.
- *No Technical Complexity:* All animation processing is handled in-app (cloud-side); the user only sees the end result. This feature differentiates the app,

leveraging emerging **image-to-video AI** capabilities that can “transform a static image into a dynamic clip in seconds”[2].

Advanced Video Enhancements: In addition to photos, the app offers creative tools for videos:

- *Add Person into Video:* A user can take a photo of someone and **integrate that person into a video** clip (e.g. add a missing family member into a family video). The app uses AI segmentation and motion matching to overlay the person’s image and, if possible, animate it within the video. (For instance, add a grandparent’s photo into a present-day video so it feels like they were there.) This is presented as a guided “Add to Video” experience: the user picks a video, picks a photo of the person to add, and the app produces a new video with that person appearing at appropriate moments.
- *Extend Video Length:* The app can take a short video and intelligently **create a slightly longer version**. For example, a 3-second old film clip could be turned into a 6-second looping or slow-motion video without obvious repeats. The system might extrapolate frames or reuse segments with smooth transitions. This helps make very short nostalgic clips more shareable (e.g., a brief clip of a baby’s first steps can be extended to give viewers a better look).
- *Video Quality Upscale:* Similar to photo enhancement, old videos (like VHS transfers) can be stabilized, color-corrected, and upscaled. A user might run a 1980s home video through the app to get reduced noise and sharper resolution. This happens via cloud processing due to the heavy computation needed.

User Library & Editing Workflow:

- *Unified Library:* The app shows a combined gallery of all photos and videos the user has imported or synced from connected sources. Users can create albums or worksets for specific projects (e.g. an album for “Family Reunion 1990” they plan to enhance). Original files from external sources are labeled with their origin (e.g. a Google Photos icon) for clarity.
- *Non-Destructive Edits:* Any enhancement or animation creates a new saved version in the app’s library. The original photo/video remains untouched (unless the user chooses to replace it when exporting). Users can compare original vs. enhanced side-by-side. They can always revert changes or try multiple different enhancements on the same original file.
- *Save & Share:* After editing, the user can **download the enhanced photo/video** to their device, or use the app’s **Share** function to post to social media or

messaging apps. This uses the native sharing APIs on mobile and the web (no separate social network in-app, but easy integration outward).

Integration Requirements

Identity and Profiles: The application uses **social sign-in** for a frictionless start. Users can log in via:

- **Microsoft Account** (e.g. @outlook.com / @live.com) – enabling easy OneDrive access.
- **Google Account** – enabling access to Google Photos.
- **Facebook Login** – primarily for identity (and potentially to fetch Facebook-hosted photos, if the API permits).

Upon first login, a user profile is created. This profile stores the user's settings and a list of connected photo repositories. Using industry-standard OAuth flows, the app requests permissions from each service the user connects (for example, the Google Photos integration will ask for read/write access to the user's library). All identity tokens are stored securely.

External Photo Repository Connectivity: Core to the product is allowing users to bring in content from where their photos already live:

- **Google Photos Integration:** Users can connect their Google Photos account and **browse their cloud library from inside the app**. They can use the app's search (which under the hood can query Google Photos via API for things like date or person) or manually pick from albums. Selected photos and videos are **imported into the app** (copied to the app's storage). Google's Picker API allows a secure selection of items[3]. After enhancement, the user can choose to **upload the new versions back to Google Photos**, so that their Google library gets the enhanced copy[4]. (For instance, an old black-and-white photo in Google Photos can be colorized in the app, and then the color version can be saved alongside the original in Google Photos.)
- **Microsoft OneDrive Integration:** Similarly, a user can connect their OneDrive (or OneDrive for Business) to access photos and videos stored there. The app will utilize Microsoft Graph APIs to list OneDrive folders (especially the Pictures folder or known albums) and let the user import items. After processing, the app can upload the enhanced media back to OneDrive (maintaining folder structure). For example, if a user picks a photo from **OneDrive > Camera Roll**, after

enhancement the updated file can be placed back into that same folder (with an edited filename or in a sub-folder for edited photos).

- **Other Sources:** “Others” could include services like **Dropbox, Apple iCloud Photos, Flickr**, or even local device storage. The PRD focuses on Google and OneDrive as primary integrations (due to their large user bases and available APIs). Support for additional sources can be added over time, following a similar import/export approach. All third-party integrations must adhere to the providers’ policies (e.g., rate limits, user consent requirements).

Continuous Sync Option: By default, users import photos on-demand. However, in the user’s profile settings, they can enable **auto-sync** for any connected service. If enabled, the app will periodically (say, daily or weekly) check the external library for new items (e.g., new photos added to Google Photos) and pre-fetch them into the app’s storage. This ensures the in-app library stays up-to-date without manual action. Users can also choose whether to auto-sync enhanced results back to the source (or always do that manually). This option appeals to power users who want the app to serve as a unified, continually updated hub of all their photos.

Data Handling and Privacy: The app **copies media to its own cloud storage** (Azure Blob) for processing, which means user content is being stored on Azure. The design must ensure:

- Secure transfer (HTTPS) for all media imports and exports.
- Proper isolation: each user’s content is kept separate and private in Blob Storage (e.g., using unique containers or directory paths per user ID).
- If a user disconnects a service or deletes their account, any personal data (photos, tokens) associated with them is purged from our storage, respecting user control.
- No external service credentials are stored in plaintext; use tokens and refresh them securely. Comply with Google’s and Facebook’s data policies when handling their user data[5].

Importantly, all the complex integration work remains **behind the scenes**. To the user, the app simply feels like it can “see” their photos on other platforms and lets them function as if all those photos were local. They are never forced to switch apps or manually download files outside our app.

Platform & Architecture Specifications

Supported Platforms (Front-end):

- **Web Application:** Accessible via modern browsers. The web UI should be responsive and offer the same core functionalities as mobile. It might be implemented as a Single Page Application (SPA) using a framework like React or Angular. The web app is ideal for users on desktop or laptop, who might take advantage of larger screens when working with many photos or doing batch enhancements.
- **iOS Application:** A native app for iPhone/iPad (iOS). This should integrate with iOS features for convenience: for instance, users can enable access to the device's local Photos app as an additional source (should they want to import an image directly from their camera roll). Use Swift/Objective-C or a cross-platform tool like Xamarin/MAUI with C# to align with backend tech.
- **Android Application:** A native Android app, offering similar integration (e.g., the ability to share a photo from the Android gallery *into* our app for enhancement). Ensure compatibility with a range of device types and Android versions. Could be built in Kotlin/Java or via cross-platform frameworks.
- **Consistency:** All platforms must offer a consistent user experience and synchronized data. A user logged in on multiple devices sees the same connected accounts and imported items (since those are tied to the cloud profile). For example, they might import photos via their phone, enhance them on the phone, and later see those results on the web app automatically.

Backend (Cloud) Architecture:

- **Azure Hosting:** The backend runs on Microsoft Azure to leverage close integration with Azure Blob Storage and scalable compute. It could be hosted as an Azure App Service or containerized and run in Azure Kubernetes Service (AKS) depending on scale needs.
- **Technology Stack:** C# (.NET Core) is the primary backend language, aligning with Azure and enabling reuse of any existing image processing libraries in .NET. The backend is essentially a set of web services (REST API endpoints) that the client apps call for various actions (search, import, enhance, etc.).
- **Azure Blob Storage:** All photo and video files handled by the app are stored here. Blob Storage is chosen for its scalability and ability to handle large binary files efficiently. Each file might be stored with metadata (in Blob metadata or in a separate Azure Table/Database) indicating user, original source, and any transformations done. Blobs also facilitate generating short-lived URLs for upload/download, which can be useful when transferring to/from client or third-party services.
- **Processing Pipeline:** Enhancements like AI colorization or animation likely utilize specialized libraries or AI models. These could run in:

  - **Azure Functions or Background Services:** For long-running tasks (video rendering, heavy AI inference), the app will offload work to background job processors. For example, when a user requests a photo animation, the request is accepted and a job is queued on an Azure Function or Azure Batch that handles the image-to-video conversion. The user is then notified (within the app) when the result is ready for viewing/download.
  - **AI/ML Services:** Depending on implementation decisions, the team might use Azure Cognitive Services (if available for image enhancements) or integrate open-source ML models. The PRD doesn't mandate how, but it requires that **all such processing is encapsulated server-side** – the user never has to, say, go to a third-party site or wait excessively on their device. This also ensures that even low-powered devices can use advanced features (since the heavy lifting is in the cloud).
- **Scalability & Performance:** The system should be designed to handle potentially millions of photos. Using Azure ensures we can auto-scale: more instances of the web service can spin up during peak times (e.g., holidays might see spikes of nostalgic photo sharing). Blob Storage can handle concurrent access. We'll implement a CDN or caching for frequently accessed images (like thumbnails in the app) to improve speed. For search queries, if needed, an index of the user's imported content can be kept to speed up text or face queries (though external services also have search APIs we can call).
- **Architecture Security:** All client-server communication is over HTTPS. We use OAuth tokens for external services as described. We also secure the user authentication and session management (possibly using something like Azure AD B2C or our own token system for the app's accounts). Regular penetration testing and compliance checks will be part of the process, given sensitive personal data (family photos) is at play.

Application Constraints:

- The UI should be **simple and friendly**, given the target users may not all be tech experts. Complex features (like the "add person to video") should be guided by wizards or clear instructions.
- The system will have to handle large file sizes (videos can be hundreds of MBs). We'll implement chunked uploading/downloading and show progress indicators to the user during transfers.
- Offline use: The mobile apps might allow some offline functionality (e.g., viewing already downloaded photos or doing local edits with device hardware if possible), but most core functions (cloud search, AI enhancements) require connectivity.

Monetization Model

The app will launch with a **freemium** pricing strategy:

- **Free Tier:** All users can download the app and use basic functionality at no cost. They can connect accounts, search/browse, and perform a limited number of enhancements for free. Specifically, each user gets to **perform 1–2 full enhancement operations at no charge** (exact number TBD, e.g., "Your first 2 enhancements are free"). This lets them try out a photo animation or colorization on their own content and see the value.
    - Additionally, features like browsing and organizing are free and unlimited. The limit only applies when invoking the *enhancement processing* (like generating an animation or doing a high-res restore), since those are resource-intensive and our key value-add.
- **Premium Usage – Pay as You Go:** For occasional users who don't want a subscription, we offer in-app purchases of enhancement credits. For example, a user can buy a pack of 10 enhancements for a few dollars. Each credit might correspond to one photo enhancement or a short video render. This micro-transaction model ensures even casual users can pay once for the specific task they need.
- **Subscription Tier:** For power users (like the family archivist who might be enhancing hundreds of photos), a subscription model provides better value. A possible offering is a **Monthly Premium Plan** that includes unlimited enhancements and priority processing (their jobs get slightly higher server priority). Subscribers might also get early access to new features. The cost of subscription and details will be determined by market research, but should be priced attractively for hobbyist archivists (e.g., $X per month).
- **Feature Gating:** All core features are available to all users, but the free tier limits how much they can use the enhancement capabilities. Some advanced features might count as multiple credits (for instance, a complex "add person to video" might consume 2 credits due to more processing). There will **not** be disruptive ads in the app – the value proposition is that the user pays with either a one-time fee or subscription for more enhancements, rather than viewing ads. We want to keep the experience focused on personal content, not ad-driven.
- **Upsell Approach:** The app should gently encourage upgrading after the free enhancements are used. For example, after a user has done their second free enhancement, when they attempt another, they can receive a prompt: *"You've used your 2 free enhancements. Unlock unlimited enhancements and keep bringing your memories to life!"* with options to buy credits or subscribe. This prompt must be clear about the cost and the benefit. Until they upgrade, they

can still use the app for browsing and managing photos – it's not locked down entirely.

- **Metrics & Adjustments:** We will track conversion rates (what percentage of users go from free to paid, and which features drive that). If, for instance, the photo-to-video animation is extremely popular, we might introduce some paywall after a certain usage (or conversely, use it as the hook to get people to subscribe). The model is flexible to adjustments post-launch, but the PRD assumption is that a *small amount of free use* is critical for initial adoption and word-of-mouth.

Priority of Implementation:
In summary, this PRD outlines a product that **focuses on core innovation (photo animation)** and solid integration. At launch, the **key deliverables** are expected to be: multi-platform apps with reliable Google Photos/OneDrive connectivity, excellent photo enhancement quality (colorize and upscale that "wow" users), and the marquee photo-to-video clip feature working smoothly (even if video-integration features are somewhat basic initially). By addressing the needs of our target users, the app aims to become the go-to solution for breathing new life into old photos, all while making it effortless to find and share those memories. Future iterations can expand on features and integrations, but the foundation laid out here is focused and attainable within a first release. Each requirement above directly ties to user value: **finding memories, enhancing them, and sharing them**, which together create a compelling reason for users to embrace the product.