Project Report

On

# PICFOLIO

## “PicFolio: Photo Management App”

Submitted in partial fulfillment of the requirement for the award of the

**DIPLOMA** in **COMPUTER ENGINEERING**

By


| | | |
|---|---|---|
| **MEET PATEL** | : | S002 |
| **KARAN DALAL** | : | S006 |
| **SHRAVANI GOLAMPALLE** | : | S042 |

Under the guidance of

**Mrs. Prachi Arora**

Lecturer, Computer Engineering Department,



Shri Vile Parle Kelavani Mandal’s

**SHRI BHAGUBHAI MAFATLAL POLYTECHNIC**


**Academic Year: 2023 - 2024**

# SHRI BHAGUBHAI MAFATLAL POLYTECHNIC

VileParle(west), Mumbai – 400 056

## PROJECT REPORT ON

Title: _____

Submitted in parallel Fulfilment of Requirements

The Diploma in Computer Engineering

Name of the Student : _____

_____

Roll No: _____

SAP No : _____

Academic Year : _____

Name and Address of the Company : _____
(If any sponsored Project/ Internship Training)

_____

_____

THIS IS TO CERTIFY THAT

Shri/ Smt. / Kum. _____

Exam Seat No. _____ Has Satisfactorily Completed

his/her Project Work and Submitted

Guide Head of the Dept. Principal

Date : _____

**SHRI BHAGUBHAI MAFATLAL POLYTECHNIC**

VileParle(West), Mumbai -400 056

## PROJECT APPROVAL SHEET

THIS IS TO CERTIFY THAT

Shri/Smt./Kum._____

_____

SAP No._____ has presented a Project Entitled

_____

In partial fulfilment of

Diploma Program in Computer Engineering / Information Technology

Same is Approved by :

1) External Examiner _____
Name and Signature

2) Internal Examiner _____
Name and Signature

Date : _____

## COMPUTER ENGINEERING DEPARTMENT

### Vision

Create a sustainable academic environment to produce highly competent computer professionals of the future

### Mission

**M1.** To expose students to latest tools and technologies in computing.
**M2.** To foster the professional development of students by providing excellence in education.
**M3.** To Adapt rapid advancements in computing by engaging students in the lifelong learning.
**M4.** To inculcate sound ethical, moral and social values amongst students for benefit of the society.

### Program Educational Objectives

The program educational objectives (PEOs) are to produce Computer professionals.

**PEO1.** Identify, design and solve computing problems by applying knowledge in Computer engineering.
**PEO2.** Promote lifelong learning by integrating academic knowledge and practical applications.
**PEO3.** Depict effective team work and practical skills for holistic development.

## PROGRAMME OUTCOMES (POs)

1. **Basic and Discipline specific knowledge:** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:** Identify and analyze well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:** Design solutions for well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing:** Apply modern engineering tools and appropriate technique to conduct standard tests and measurements.
5. **Engineering practices for society, sustainability and environment:** Apply appropriate technology in context of society, sustainability, environment and ethical practices.
6. **Project Management:** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
7. **Life-long learning:** Ability to analyze individual needs and engage in updating in the context of technological changes

## PROGRAMME SPECIFIC OUTCOMES (PSOs)

PSO1: Demonstrate the fundamental knowledge in the areas of Operating system, Web Technology, Microprocessor based system and IOT by applying programming skills and developing applications.

PSO2: Administer and manage Open source, Networking, Security and Database domains to enhance student growth.

# Abstract

PicFolio is a comprehensive photo management application designed to streamline the organization and access of personal photo collections across multiple devices. With PicFolio, users can seamlessly sync their mobile devices with their PCs, enabling effortless uploading and viewing of photos. The application simplifies the process by providing users with a unique code and designated directory for storing photos on their PCs, ensuring easy accessibility and management.

In addition to basic photo storage and retrieval functions, PicFolio incorporates advanced features such as machine learning algorithms for face clustering and image tagging. These algorithms automatically categorize photos based on detected faces and apply relevant tags, enhancing the organization and searchability of the photo library. Users can effortlessly group photos into albums, facilitating personalized organization and navigation.

PicFolio prioritizes user convenience and security by implementing user accounts, allowing individuals to securely access and manage their own photo collections. Moreover, the application includes a sharing space, enabling users to effortlessly share photos with friends and family. By combining advanced technology with user-friendly design, PicFolio revolutionizes the way users manage and interact with their personal photo collections, making photo management a seamless and enjoyable experience.

# List Of Contents

# Chapter 1

# Introduction

## 1.1 Introduction

PicFolio is an innovative photo management application designed to revolutionize the way users interact with their image libraries. By seamlessly integrating both mobile and desktop platforms, PicFolio offers a comprehensive solution for organizing, accessing, and sharing photos effortlessly. With the PicFolio mobile application installed on your smartphone and the corresponding software on your PC, users gain access to a unified ecosystem where they can efficiently manage their photo collections.

The core functionality of PicFolio centers around its intuitive interface, empowering users to store, organize, and view their photos with ease. Upon installation, the software generates a unique code and designated directory for storing photos on the user's PC, establishing a seamless connection between their mobile device and computer. Leveraging advanced machine learning algorithms such as face clustering and image tagging, PicFolio automates the process of organizing photos, ensuring that images are appropriately tagged and categorized based on content. This intelligent feature not only streamlines the management of photo libraries but also enhances user experience by minimizing manual intervention. Additionally, PicFolio incorporates user accounts, enabling individuals to securely access and manage their own photos while also providing a sharing platform for distributing images with friends and family. With PicFolio, users can enjoy a photo management experience that optimizes organization, accessibility, and sharing capabilities.

## 1.2 Problem Statement

Problem Statement: Seamless Integration for Photo Management and Accessibility

The current landscape of digital photo management is plagued by fragmentation and inefficiency, prompting the need for a holistic solution that bridges the gap between mobile devices and PCs. The following challenges underscore the urgency for a comprehensive photo management application:

1. Fragmented Ecosystem: With users capturing and storing photos across various devices, including smartphones, tablets, and PCs, there exists a fragmented approach to managing these digital assets. This fragmentation leads to disjointed user experiences, as users struggle to synchronize and organize their photo collections seamlessly.

2. Manual Synchronization Hassles: The absence of an integrated solution results in users resorting to manual methods for transferring photos between their mobile devices and PCs. This manual synchronization process is not only time-consuming but also prone to errors, leading to inconsistencies and redundancies in photo libraries.

3. Tedious Organization and Tagging: Traditional photo management tools often lack intelligent features for automated organization and tagging. Consequently, users spend considerable time manually sorting through their photo collections and applying tags, diminishing productivity and user satisfaction.

4. Accessibility and Collaboration Limitations: Existing photo management solutions fall short in facilitating seamless photo accessibility and collaboration among multiple users. This limitation inhibits users from effortlessly sharing and collaborating on photos, restricting the utility and versatility of their digital assets.

5. Privacy and Security Vulnerabilities: In an era marked by heightened concerns over data privacy and security, many photo management applications fail to provide robust safeguards against unauthorized access and data breaches. Without adequate security measures in place, users risk compromising their sensitive personal data.

## 1.3 Proposed Solution

Proposed Solution: Photo Management Application

Addressing the challenges outlined in the problem statement necessitates the development of a comprehensive photo management application that seamlessly integrates with mobile devices and PCs. This has the following key components:

1. Automated Synchronization Mechanism: Leveraging cloud-based storage and synchronization technology, the application will offer seamless synchronization of photos across all linked devices. Changes made to the photo library on one device will be automatically propagated to other devices in real-time, ensuring synchronization without manual intervention.

2. Intelligent Organization and Tagging: Advanced machine learning algorithms will be employed to automate the organization and tagging of photos. Utilizing techniques such as image recognition and natural language processing, the application will intelligently categorize photos based on content, and other relevant metadata, thereby simplifying the process of organizing photos.

3. Enhanced Accessibility and Collaboration Features: The application will facilitate effortless sharing and collaboration on photos among multiple users. Users will have granular control over sharing permissions, allowing them to selectively share individual photos, with specific individuals or groups. Collaborative editing features will enable users to annotate, comment on, and edit shared photos collaboratively in real-time.

4. Robust Security and Privacy Measures: Security and privacy will be paramount considerations in the design and implementation of the application. End-to-end encryption will be employed to protect users' photos and data during transmission and storage. Additionally, robust authentication mechanisms, such as biometric authentication and two-factor authentication, will safeguard users' accounts against unauthorized access.

# Chapter 2

# Review of Literature

## 2.1 Technical Paper Review

We present the Recognize Anything Model (RAM): a strong foundation model for image tagging. RAM makes a substantial step for large models in computer vision, demonstrating the zero-shot ability to recognize any common category with high accuracy. RAM introduces a new paradigm for image tagging, leveraging large-scale imagetext pairs for training instead of manual annotations. The development of RAM comprises four key steps. Firstly, annotation-free image tags are obtained at scale through automatic text semantic parsing. Subsequently, a preliminary model is trained for automatic annotation by unifying the caption and tagging tasks, supervised by the original texts and parsed tags, respectively. Thirdly, a data engine is employed to generate additional annotations and clean incorrect ones. Lastly, the model is retrained with the processed data and fine-tuned using a smaller but higherquality dataset. We evaluate the tagging capabilities of RAM on numerous benchmarks and observe impressive zero-shot performance, significantly outperforming CLIP and BLIP. Remarkably, RAM even surpasses the fully supervised manners and exhibits competitive performance with the Google tagging API. We are releasing the RAM at https: //recognize-anything.github.io/ to foster the advancements of large models in computer vision.

Large language models (LLM) trained on large-scale web datasets have sparked a revolution in nature language processing (NLP). These models[20, 5] exhibit impressive zero-shot generalization, enabling them to generalize to tasks and data distributions beyond their training domain. When it comes to computer vision (CV), Segment Anything Model (SAM) [12] has also demonstrated remarkable zeroshot localization abilities through data scaling-up.

Multi-label image recognition, also known as image tagging, aims to provide semantic labels by recognizing multiple labels of a given image. Image tagging is a significant and practical computer vision task, as images inherently contain multiple labels encompassing objects, scenes, attributes, and actions. Regrettably, existing models in multi-label classification, detection, segmentation, and vision-language approaches have exhibited deficiency in tagging, characterized by limited scopes or poor accuracy, as illustrated in Figure 1. Two core components impede the progress of image tagging. 1) The difficulty lies in collecting large-scale highquality data. Specifically, there is a lack of a universal and unified label system and an efficient data annotation engine, capable of semi-automatic or even automatic annotation of large-scale images with a vast number of categories. 2) There is a lack of efficient and flexible model design that can leverage large-scale weakly-supervised data to construct an open-vocabulary and powerful model.

Label System: We begin by establishing a universal and unified label system. We incorporate categories from popular academic datasets (classification, detection, and segmentation) as well as commercial tagging products (Google, Microsoft, Apple). Our label system is obtained by merging all the public tags with the common tags from texts, thus covering most of common labels with a moderate amount of 6,449. The remaining open-vocabulary labels can be identified through open-set recognition.

Dataset: How to automatically annotate large-scale images with the label system is another challenge [30]. Drawing inspiration from CLIP [22] and ALIGN [11], which leverage publicly available image-text pairs at scale to train powerful visual models, we adopt similar datasets for image tagging. To utilize these large-scale image-text data for tagging, following [9, 10], we parse the texts and obtain the image tags through automatic text semantic parsing. This process allows us to obtain a diverse collection of annotation-free image tags in accordance with image-text pairs.

Data Engine: However, the image-text pairs from the web are inherently noisy, often containing missing or incorrect labels. To enhance the quality of annotations, we design a tagging data engine. In addressing missing label, we leverage existing models to generate additional tags. With regards to incorrect labels, we first localize specific regions corresponding to different tags within the image. Subsequently, we employ region clustering techniques to identify and eliminate outliers within the same class. Furthermore, we filter out tags that exhibit contrary predictions between whole images and their corresponding regions, ensuring a cleaner and more accurate annotation.

Training Phase. RAM is pretrained on large-scale datasets with a resolution of 224 and fine-tuning at a resolution of 384 using small and high-quality datasets. Empirical evidence suggests that RAM converges rapidly, often with convergence achieved after a minimal number of epochs (typically less than 5 epochs). This accelerated convergence enhances the reproducibility of RAM with limited computational resources. To illustrate, the version of RAM pretrained on 4 million necessitate 1-day of computation, and the strongest version of RAM pretrained on 14 million images necessitate a mere 3-days of computation on 8 A100 GPUs.

Inference Phase. The lightweight image-tag recognition decoder effectively ensures the inference efficiency of RAM on image tagging. Furthermore, we eliminate the self-attention layers from the recognition decoder, which only further improves efficiency but also circumvents potential interference between label queries. Consequently, instead of fixed categories and quantities, RAM allows customization of label queries for any category and quantity which want to automatically recognize, enhancing its utility across various visual tasks and datasets.

References

[1] Apple Developer. https://developer.apple.com/ documentation/vision.

[2] Google Cloud vision API. https://cloud.google. com/vision.

[3] Microsoft Azure cognitive service. https://azure.microsoft.com/zh-cn/products/cognitive

# Chapter 3

# Report on Present Investigation

## 3.1 Flutter Framework:

### 3.1.1 Introduction to Flutter:

Flutter is an open-source UI software development kit (SDK) that offers a comprehensive toolkit for creating natively compiled applications for mobile, web, and desktop platforms from a single codebase. Built upon the Dart programming language, Flutter provides developers with a powerful and expressive framework to design visually stunning interfaces and deliver engaging user experiences.

One of the defining features of Flutter is its emphasis on a reactive and declarative programming paradigm, enabling developers to efficiently express the user interface layout and behavior in a concise and intuitive manner. With Flutter's widget-based architecture, developers can compose complex UI elements using a rich library of customizable widgets, allowing for seamless integration of design elements and interactions.

Furthermore, Flutter's hot reload feature revolutionizes the development workflow by enabling developers to instantly see the effects of code changes reflected in the running application, without the need for time-consuming recompilation or application restarts. This iterative and rapid development process fosters experimentation, iteration, and collaboration, ultimately accelerating the pace of development and enhancing productivity.

Beyond its development-centric features, Flutter also excels in delivering native performance across multiple platforms. By compiling Dart code to native machine code, Flutter ensures that applications run smoothly and efficiently, with fast rendering, responsive animations, and fluid user interactions. This native performance is complemented by Flutter's extensive support for platform-specific features and

APIs, enabling developers to seamlessly integrate platform-specific functionalities into their applications.

Overall, Flutter represents a paradigm shift in the world of app development, offering developers a powerful and versatile framework for building cross-platform applications with unparalleled speed, flexibility, and performance. With its expressive UI toolkit, seamless hot reload feature, and native performance capabilities, Flutter is poised to reshape the future of app development and empower developers to bring their creative visions to life on a global scale.

### 3.1.2 Features of Flutter Framework:

**Cross-Platform Development**: Flutter allows developers to write code once and deploy it across multiple platforms including iOS, Android, web, and desktop, using a single codebase. This significantly reduces development time and effort, as developers can target a wider audience with minimal additional work.

**Expressive UI**: Flutter provides a rich set of customizable widgets and tools for creating visually appealing and highly interactive user interfaces. Its flexible UI framework empowers developers to implement complex designs with ease, enabling the creation of delightful user experiences.

**Hot Reload**: One of Flutter's standout features is its Hot Reload capability, which allows developers to instantly see changes made to the code reflected in the app while it's running. This enables rapid experimentation, debugging, and iteration, resulting in faster development cycles and increased productivity.

**Native Performance**: Flutter delivers native-like performance by compiling code directly to native ARM machine code, ensuring smooth animations, fluid transitions, and responsive user interactions. This results in high-performance applications that run seamlessly on each platform.

**Access to Native Features**: Flutter offers extensive support for accessing platform-specific features and APIs, allowing developers to integrate functionalities such as camera, location, sensors, and more seamlessly into their applications. This enables the creation of feature-rich and highly functional applications tailored to the needs of users.

**Strong Community Support**: Flutter boasts a vibrant and growing community of developers, contributors, and enthusiasts who actively contribute to its development, share knowledge, and provide support. This collaborative ecosystem fosters innovation, accelerates learning, and ensures the continuous improvement of the framework.

**Stateful Hot Reload**: In addition to traditional hot reload, Flutter also supports stateful hot reload, which preserves the state of the app during code changes. This allows developers to iterate on UI changes without losing the current app state, providing a smoother development experience.

**Modular Architecture**: Flutter's modular architecture and reactive programming paradigm make it easy to organize code into reusable components, enhancing code maintainability and scalability. Developers can build UI components as independent widgets, making it easier to manage and update the application's interface.

**Internationalization and Accessibility**: Flutter provides robust support for internationalization and accessibility, allowing developers to create applications that cater to diverse audiences and comply with accessibility standards. This includes support for localization, RTL (right-to-left) languages, screen reader compatibility, and more.

**Comprehensive Tooling**: Flutter comes with a comprehensive set of development tools, including the Flutter SDK, command-line interface (CLI), and integrated development environments (IDEs) such as Visual Studio Code and Android Studio. These tools streamline the development process, providing features such as code autocompletion, debugging, and performance profiling.

**Huge Widget Library**: The reason why Flutter app developers can build applications faster is because of its impeccable ready-to-use widget collection. Along with a huge collection of widgets, it also has animations that you can select to make your application interactive and engaging.

**Open-Source**: Flutter is a free and open-source framework for developing applications, which means that developers can use it for free and contribute to its development.

### 3.1.3 Flutter Architecture

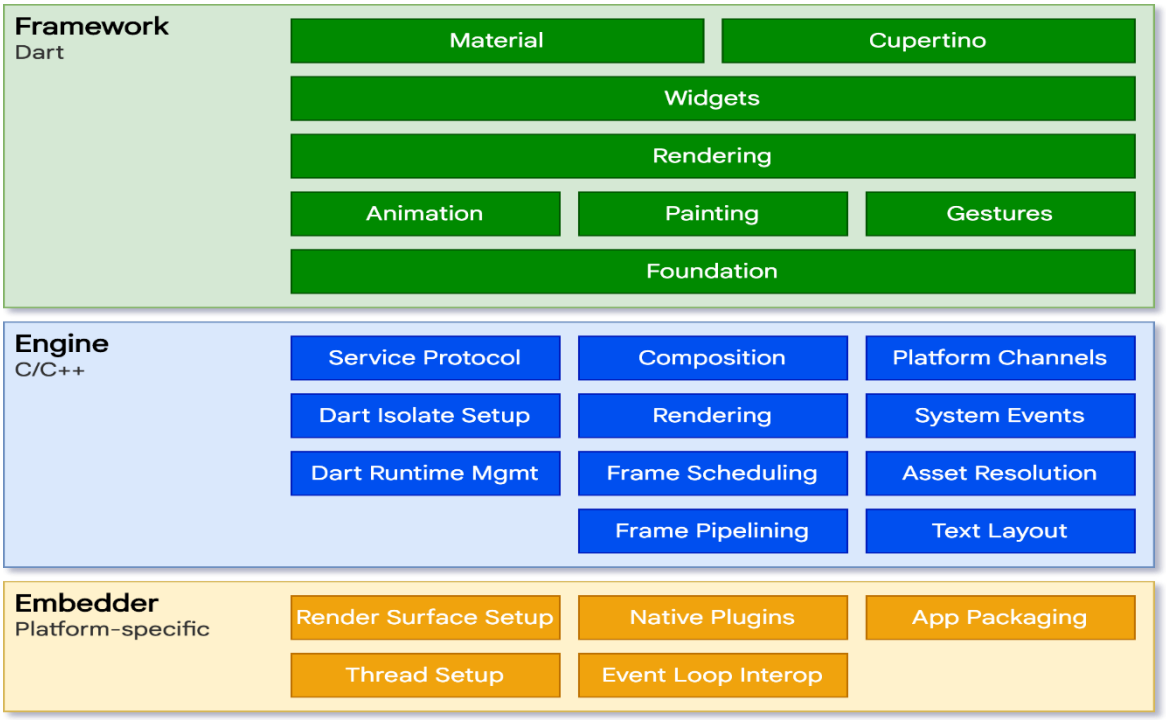



Fig1 : Flutter Architecture

## 3.2 Visual Studio

### 3.2.1 Introduction to Visual Studio:

Visual Studio is an integrated development environment (IDE) developed by Microsoft. It is used for developing software applications, websites, and mobile apps for Windows, iOS, and Android. Visual Studio supports multiple programming languages, including C++, C#, JavaScript, and Python.

These are key features of Visual Studio:

**Code Editor**: Visual Studio provides a powerful code editor with features like syntax highlighting, code completion, and error checking.

**Debugging**: Visual Studio has a built-in debugger that allows developers to debug their code in real-time and identify and fix errors.

**Intellisense**: Visual Studio includes Intellisense, which provides suggestions and auto- completion for code as developers type.

**Refactoring**: Visual Studio has built-in refactoring tools that make it easy to rename variables, extract methods, and perform other code refactorings.

**Version Control**: Visual Studio includes support for version control systems like Git and Team Foundation Server (TFS), making it easy to manage code changes.

**Project Templates**: Visual Studio includes project templates for a variety of programming languages and frameworks, making it easy to get started with new projects.

**Code Snippets**: Visual Studio includes a library of code snippets, which can be easily inserted into code to perform common tasks.

**Extensions**: Visual Studio supports extensions, which can be used to add new features and functionality to the IDE.

**Testing**: Visual Studio includes built-in testing tools, which allow developers to create and run automated tests for their code.

**Cross-Platform Development**: Visual Studio supports cross-platform development, allowing developers to create applications for Windows, iOS, and Android using a single codebase.

## 3.3 SQLite

### 3.3.1 Introduction to SQLite:

SQLite is a lightweight, self-contained, serverless, and open-source relational database management system (RDBMS) that is widely used in various applications and platforms. Unlike traditional client-server databases, SQLite is embedded directly into the application, making it ideal for scenarios where a full-fledged database server is not required or practical. SQLite is written in the C programming language and is designed to be fast, reliable, and easy to use. It operates on a single disk file and implements a transactional SQL database engine without the need for configuration or administration.

SQLite is a versatile and reliable relational database management system that

offers the benefits of simplicity, efficiency, portability, and scalability. Its lightweight design, embedded nature, SQL compatibility, and cross-platform support make it an ideal choice for a wide range of applications, from mobile apps to desktop software to embedded systems. Whether you're building a small personal project or a large-scale enterprise application, SQLite provides a solid foundation for managing and storing data effectively.

### 3.3.2 Services of SQLite:

**Documentation and Resources**: SQLite offers comprehensive documentation, tutorials, and resources to help developers learn and use SQLite effectively. This includes official documentation, user guides, tutorials, and community forums where developers can ask questions and seek help.

**Tools and Utilities**: SQLite provides a variety of tools and utilities to assist developers in managing and working with SQLite databases. This includes command-line tools, GUI-based database management applications, and libraries for integrating SQLite with programming languages and frameworks.

**Professional Support**: SQLite offers professional support services for organizations and developers who require assistance with using SQLite in their projects. This may include troubleshooting database issues, optimizing database performance, designing database schemas, and addressing security concerns.

**Commercial Licensing**: While SQLite is open-source and free to use under the public domain, SQLite also offers commercial licensing options for organizations that require additional rights and support beyond what is provided by the open-source license. Commercial licensing may include features such as legal indemnification, extended support, and custom development services.

## 3.4 Python

### 3.4.1 Introduction

Python is a versatile and dynamic programming language that has garnered widespread acclaim since its inception in the late 1980s. Developed by Guido van Rossum and first released in 1991, Python has evolved into one of the most popular languages in both the academic and professional realms. Its simplicity, readability, and extensive libraries make it an ideal choice for a wide range of applications, from web development and data analysis to artificial intelligence and scientific computing.

One of the key features that sets Python apart is its elegant syntax, which emphasizes readability and reduces the cost of program maintenance. Python code is often described as resembling executable pseudocode, making it accessible even to those with minimal programming experience. This simplicity, combined with its powerful capabilities, has contributed to Python's widespread adoption across various industries and disciplines.

Python's extensive standard library provides developers with a rich set of tools and modules for accomplishing common tasks without the need for additional downloads or dependencies. From data manipulation with pandas to web development with Django, Python's standard library covers a broad spectrum of functionalities, empowering developers to build robust and efficient solutions with minimal effort.

Moreover, Python's open-source nature and vibrant community have fostered a culture of collaboration and innovation. The Python Package Index (PyPI) hosts over 300,000 packages contributed by developers worldwide, covering virtually every conceivable application domain. This vast ecosystem ensures that developers have access to a wealth of resources and solutions to tackle any challenge they may encounter

In summary, Python's simplicity, versatility, and extensive ecosystem have cemented its position as one of the preeminent programming languages of the 21st century. Whether you're a beginner learning to code for the first time or a seasoned developer working on complex projects, Python offers a powerful yet accessible platform for turning ideas into reality.

### 3.4.2 Python Features

1. **Free and Open Source:**

   Python language is freely available at the official website. Since it is open-source, this means that source code is also available to the public. So you can download it, use it as well as share it.

2. **Easy to code:**

   Python is a high-level programming language. Python is very easy to learn the language as compared to other languages like C, C#, Javascript, Java, etc. It is very easy to code in the Python language and anybody can learn Python basics in a few hours or days. It is also a developer-friendly language.

3. **Easy to Read:**

   As you will see, learning Python is quite simple. As was already established, Python's syntax is really straightforward. The code block is defined by the indentations rather than by semicolons or brackets.

4. **Object-Oriented and Functional Programming:** Python supports both object-oriented and functional programming paradigms, allowing developers to write code in a style that best suits their needs. This flexibility enables developers to design elegant and scalable solutions, leveraging concepts such as classes, inheritance, encapsulation, polymorphism, and higher-order functions.

5. **GUI Programming Support:** Graphical User interfaces can be made using a module such as PyQt5, PyQt4, wxPython, or Tk in Python. PyQt5 is the most popular option for creating graphical apps with Python.

6. **High-Level Language:** Python is a high-level language. When we write programs in Python, we do not need to remember the system architecture, nor do we need to manage the memory.

7. **Large Community Support:** Python has gained popularity over the years. Our questions are constantly answered by the enormous StackOverflow community. These websites have already provided answers to many questions about Python, so Python users can consult them as needed.

8. **Easy to Debug:** Excellent information for mistake tracing. You will be able to

quickly identify and correct the majority of your program's issues once you understand how to interpret Python's error traces. Simply by glancing at the code, you can determine what it is designed to perform.

9. **Python is a Portable language:** Python language is also a portable language. For example, if we have Python code for Windows and if we want to run this code on other platforms such as Linux, Unix, and Mac then we do not need to change it, we can run this code on any platform.
10. **Python is an Integrated language:** Python is also an Integrated language because we can easily integrate Python with other languages like C, C++, etc.
11. **Interpreted Language:** Python is an Interpreted Language because Python code is executed line by line at a time. like other languages C, C++, Java, etc. there is no need to compile Python code this makes it easier to debug our code. The source code of Python is converted into an immediate form called bytecode.
12. **Large Standard Library :** Python has a large standard library that provides a rich set of modules and functions so you do not have to write your own code for every single thing. There are many libraries present in Python such as regular expressions, unit-testing, web browsers, etc.
13. **Dynamic Typing and Duck Typing:** Python uses dynamic typing, meaning that variable types are determined at runtime. This flexibility allows developers to write more expressive and concise code without explicitly specifying variable types. Additionally, Python follows the principle of "duck typing," which focuses on an object's behavior rather than its type, promoting code reusability and flexibility.
14. **Frontend and backend development:** With a new project py script, you can run and write Python codes in HTML with the help of some simple tags <py-script>, <py-env>, etc. This will help you do frontend development work in Python like javascript. Backend is the strong forte of Python it's extensively used for this work cause of its frameworks like Django and Flask.
15. **15. Allocating Memory Dynamically:** In Python, the variable data type does not need to be specified. The memory is automatically allocated to a variable at runtime when it is given a value

## 3.5 ML Models

### 3.5.1 Introduction to Deep Face

Deepface is a lightweight face recognition and facial attribute analysis (age, gender, emotion and race) framework for python. It is a hybrid face recognition framework wrapping state-of-the-art models: VGG-Face, Google FaceNet, OpenFace, Facebook DeepFace, DeepID, ArcFace, Dlib, SFace and GhostFaceNet.

Facial Recognition

A modern face recognition pipeline consists of 5 common stages: detect, align, normalize, represent and verify.

Face recognition requires applying face verification many times. Herein, deepface has an out-of-the-box find function to handle this action. It's going to look for the identity of input image in the database path and it will return list of pandas data frame as output. Meanwhile, facial embeddings of the facial database are stored in a pickle file to be searched faster in next time. Result is going to be the size of faces appearing in the source image. Besides, target images in the database can have many faces as well.

### 3.5.2 Recognize Anything Model

RAM is an image tagging model, which can recognize any common category with high accuracy. RAM is accepted at CVPR 2024 Multimodal Foundation Models Workshop.

Superior Image Recognition Capability

RAM++ outperforms existing SOTA image fundamental recognition models on common tag categories, uncommon tag categories, and human-object interaction phrases.

Comparison of zero-shot image recognition performance.

Strong Visual Semantic Analysis

We have combined Tag2Text and RAM with localization models (Grounding-DINO and SAM) and developed a strong visual semantic analysis pipeline in the Grounded-SAM project.

## 3.6 Hardware Requirements

### 3.6.1 For Mobile users

- The user should have access to a mobile compatible with mainstream mobile operating systems such as Android and iOS.
- Operating System: Windows 10, macOS, or Linux (latest versions).
- Web Browser: Google Chrome, Mozilla Firefox, Safari, or Microsoft Edge (latest versions).
- Processor: Intel Core i5 or AMD Ryzen 5 (or higher) with a clock speed of 2 GHz orhigher.
- Memory (RAM): A minimum of 2GB RAM is recommended to ensure smooth performance, especially when dealing with larger photo and video libraries.Graphics Card: Dedicated graphics card with at least 2 GB VRAM. Recommended options include Nvidia GeForce GTX series or AMD Radeon RX series.
- Storage: While the app itself may not consume much storage space, users typically require enough storage capacity on their device to store their photo and video collections.
- Camera: Access to a camera may be necessary for certain functionalities like capturing new photos or videos directly within the app.
- Internet Connectivity: While the app may primarily function offline, certain features like shared pages or cloud backup options may require internet connectivity.

### 3.6.2 For Development:

- Microsoft Windows 7/8/10/11 (64-bit).
- 8GB RAM minimum.
- 2GB of available disk space minimum.
- i5 Processor.
- Android Studio
- Flutter SDK

## 3.7 Software Requirements

- Visual Studio (Version 2021.1.1).
- Backend:

-Python version 8+

- Frontend:

- Flutter with Vscode

## 3.8 Diagrams and Flowcharts

### 3.8.1 Use Case Diagram:

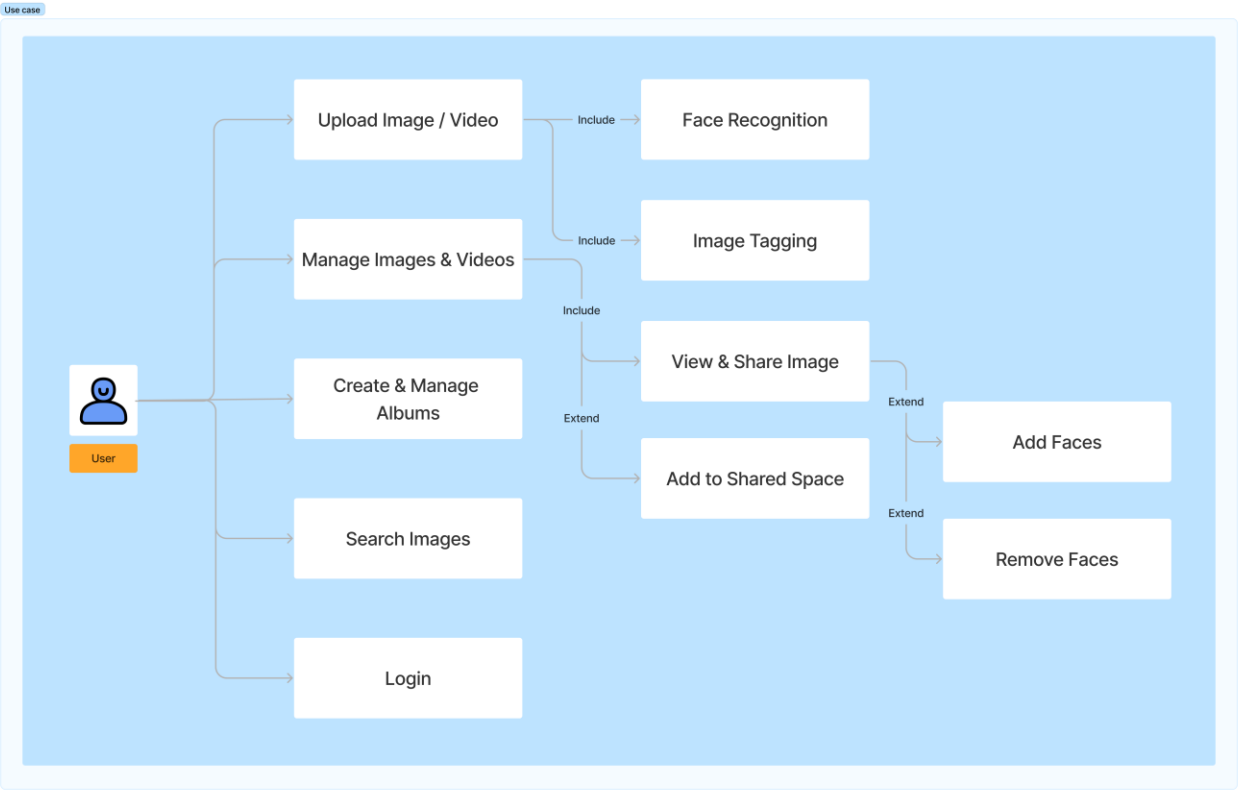



**Fig2 :** Use Case Diagram

### 3.8.2 Working Flowchart:

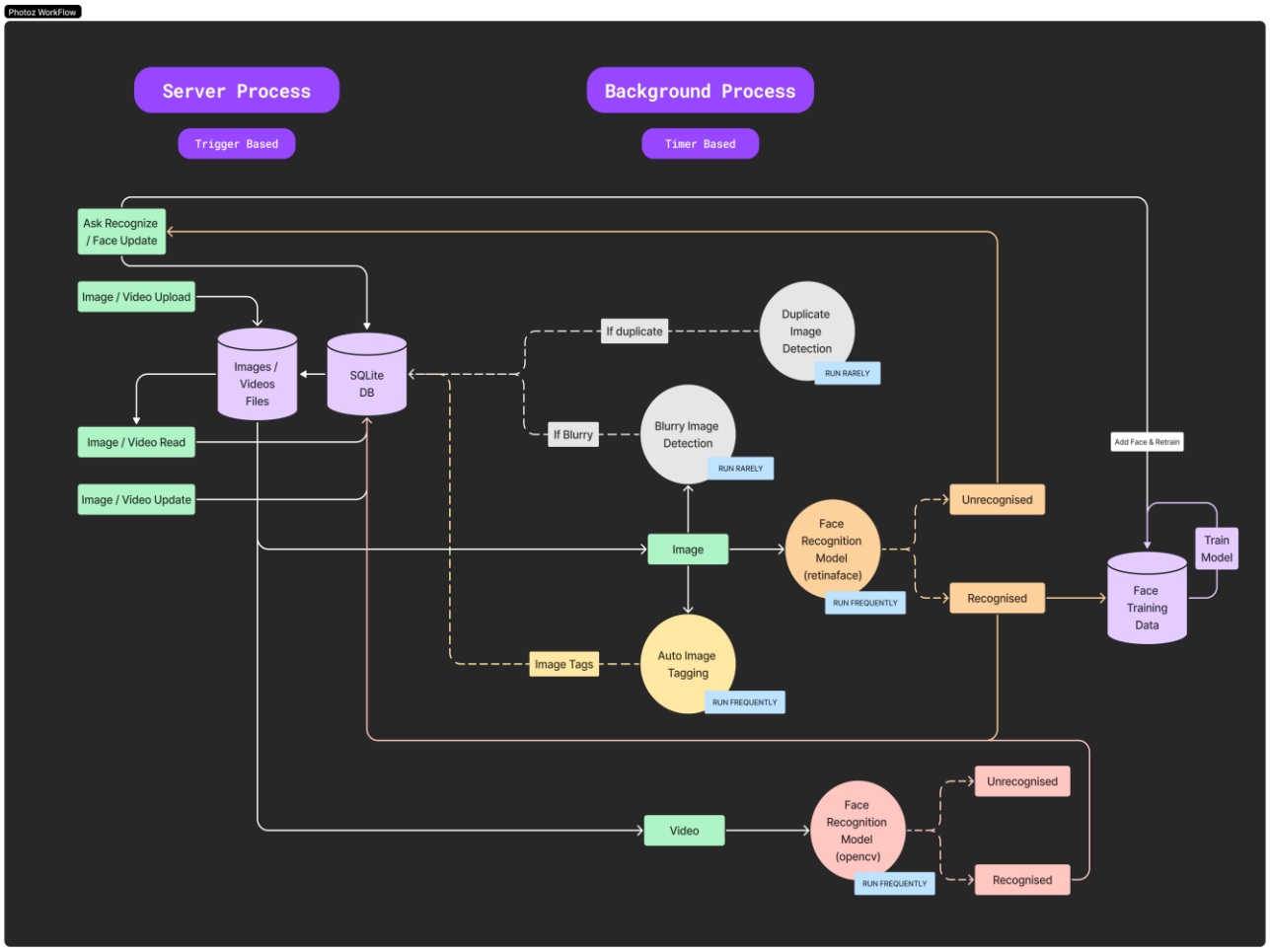



**Fig3:** Working Flowchart

### 3.8.3 Background Tasks Flow:

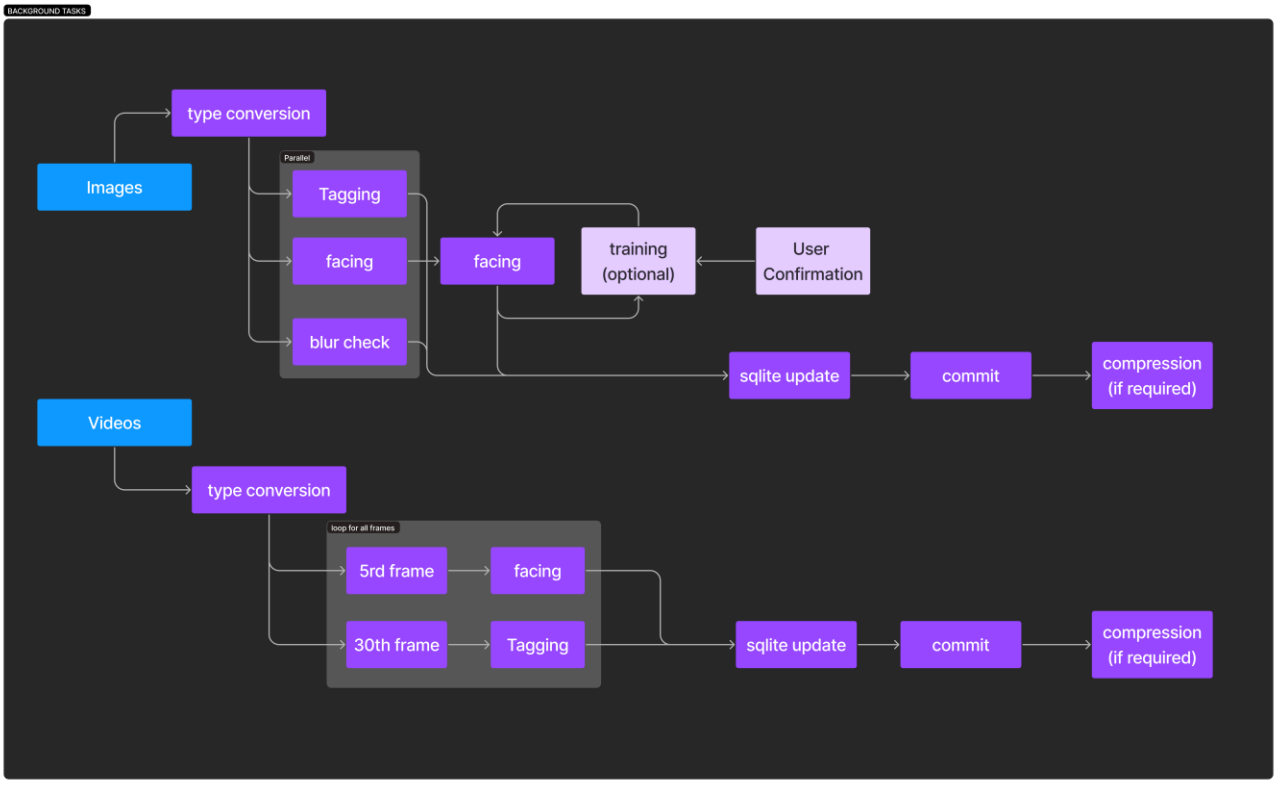



**Fig4:** Background tasks flow

### 3.8.4 Files Storage:

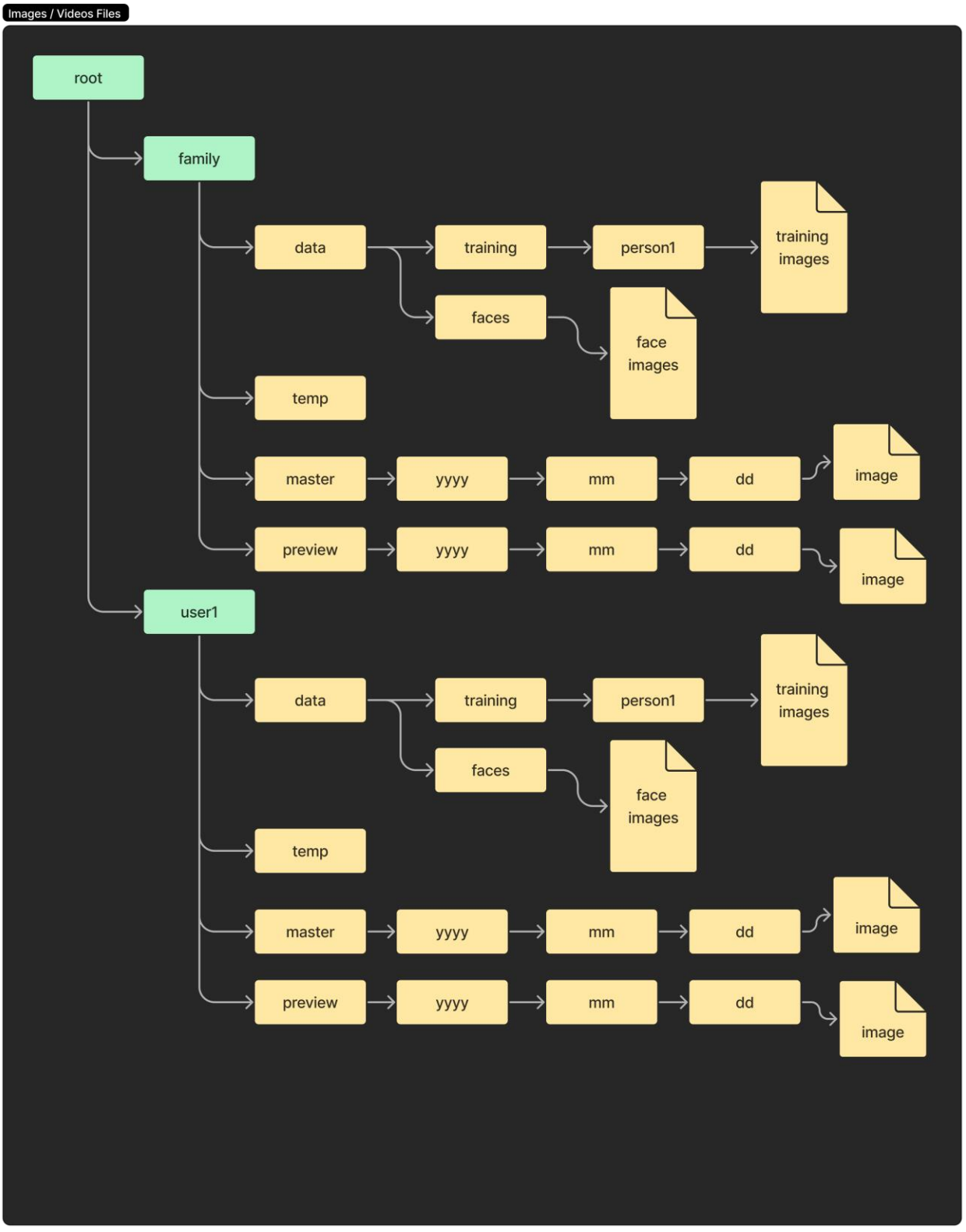



**Fig5:** Files Storage

### 3.8.5 User/login Flowchart:

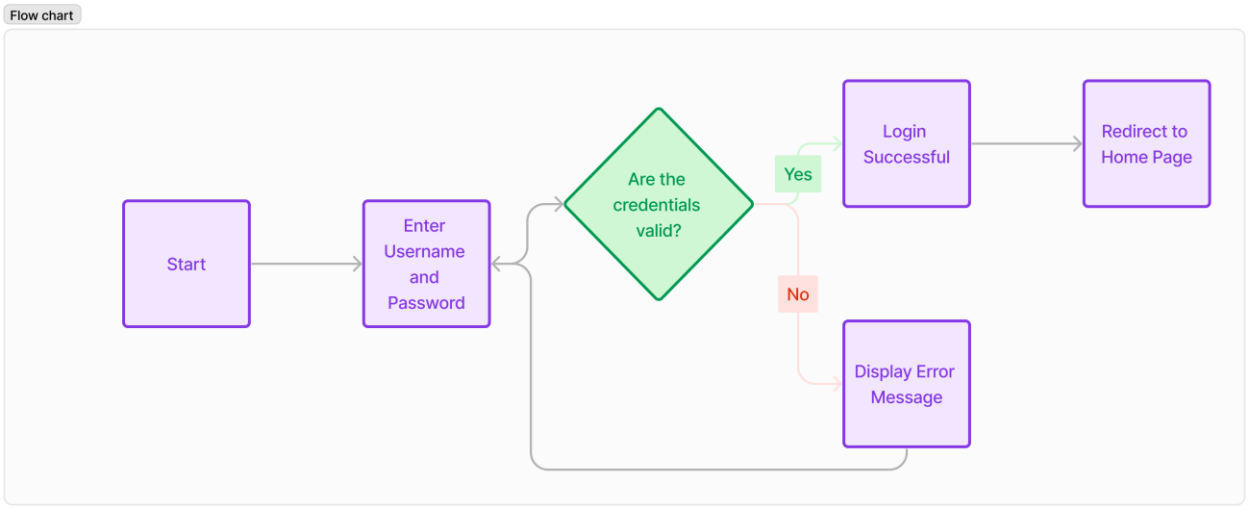



**Fig6:** User/login Flowchart

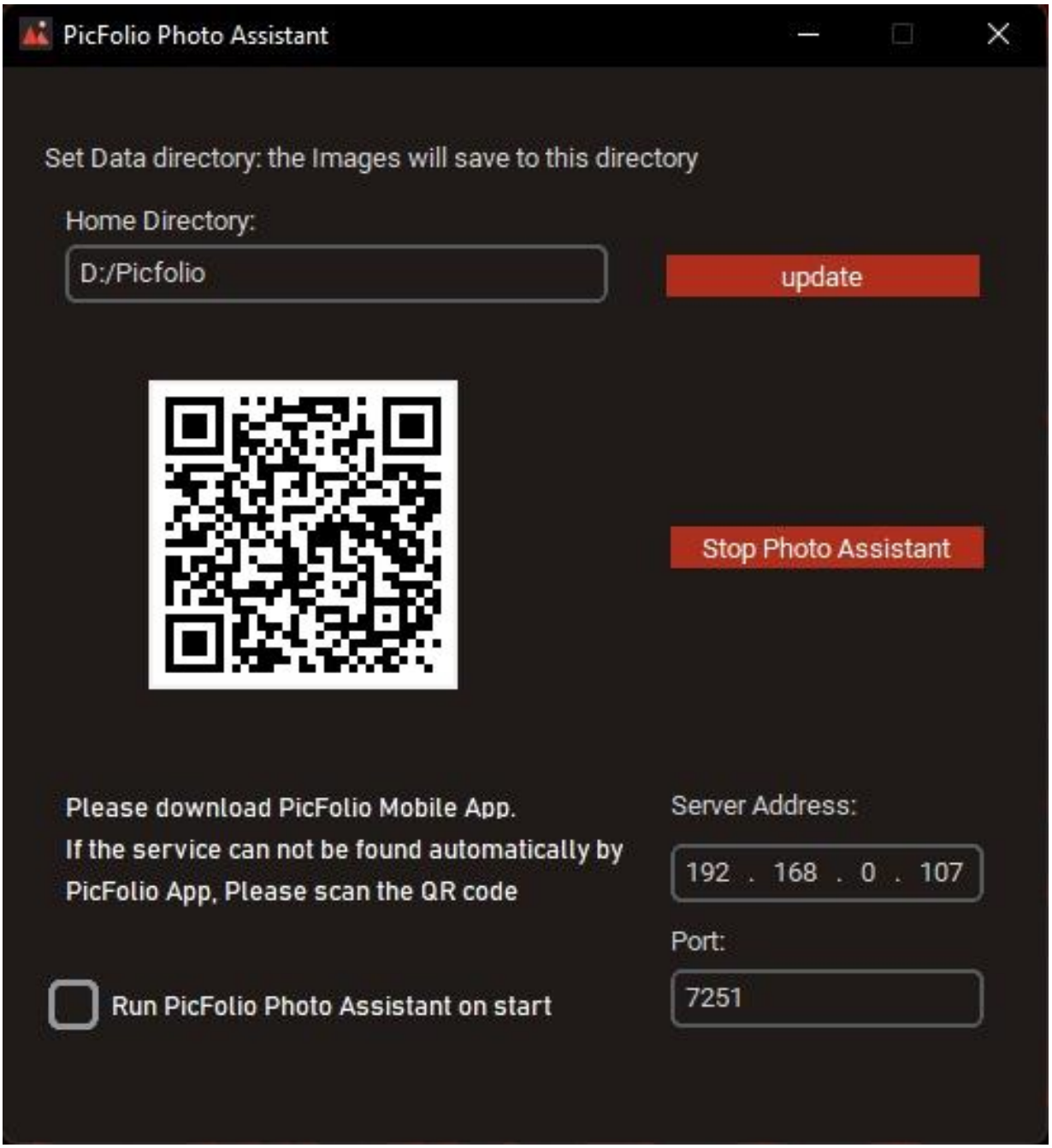
### 3.8.6 Gantt Chart of the Project:

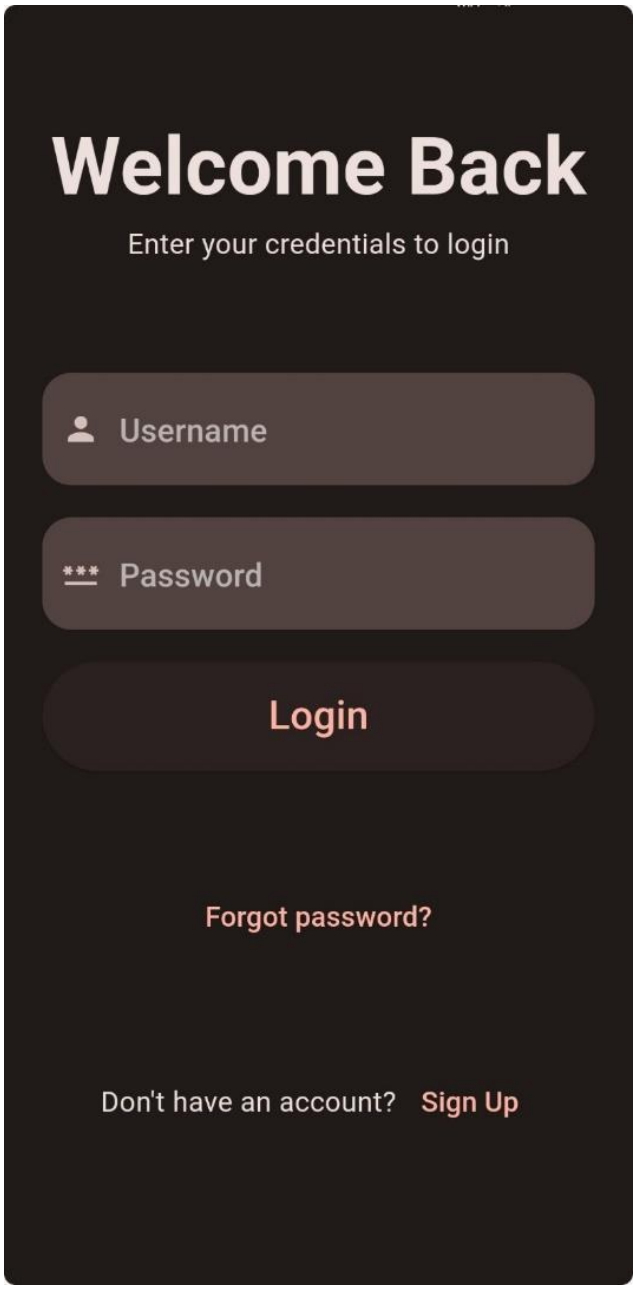
| | JANUARY | | | | FEBRUARY | | | | MARCH | | | | APRIL | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WEEKS ACTIVITIES | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| Decide the Goal of the project | | | | | | | | | | | | | | | | |
| Name and approach | | | | | | | | | | | | | | | | |
| Setting up the project environment. | | | | | | | | | | | | | | | | |
| Making of Project | | | | | | | | | | | | | | | | |
| Testing of the Project | | | | | | | | | | | | | | | | |
| Making the Required Changes | | | | | | | | | | | | | | | | |
| Deployment of Project | | | | | | | | | | | | | | | | |

## Chapter 4

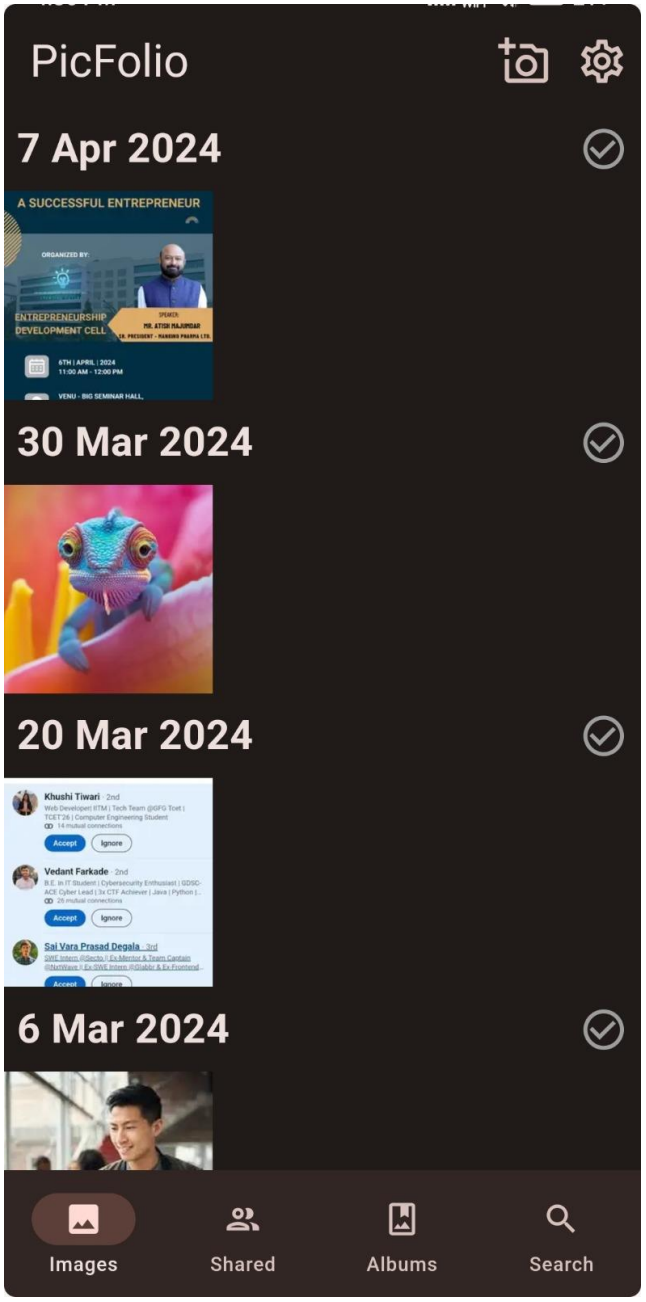
# Implementation

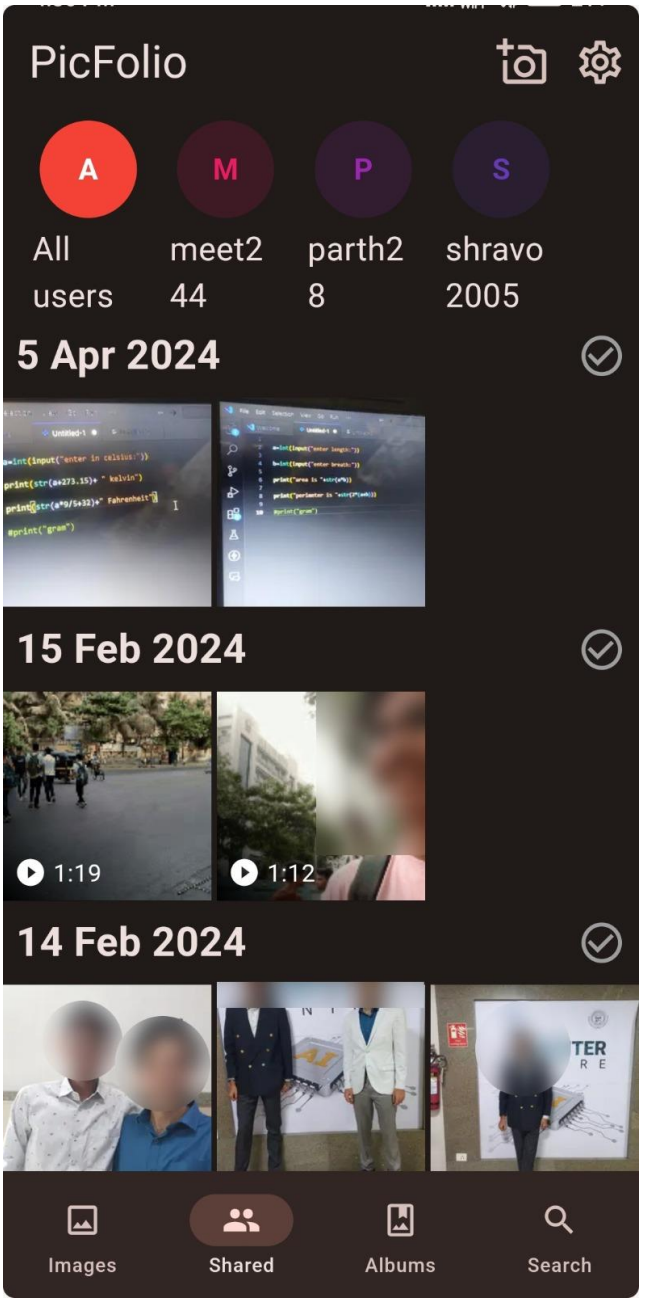
## 4.1 User Interface

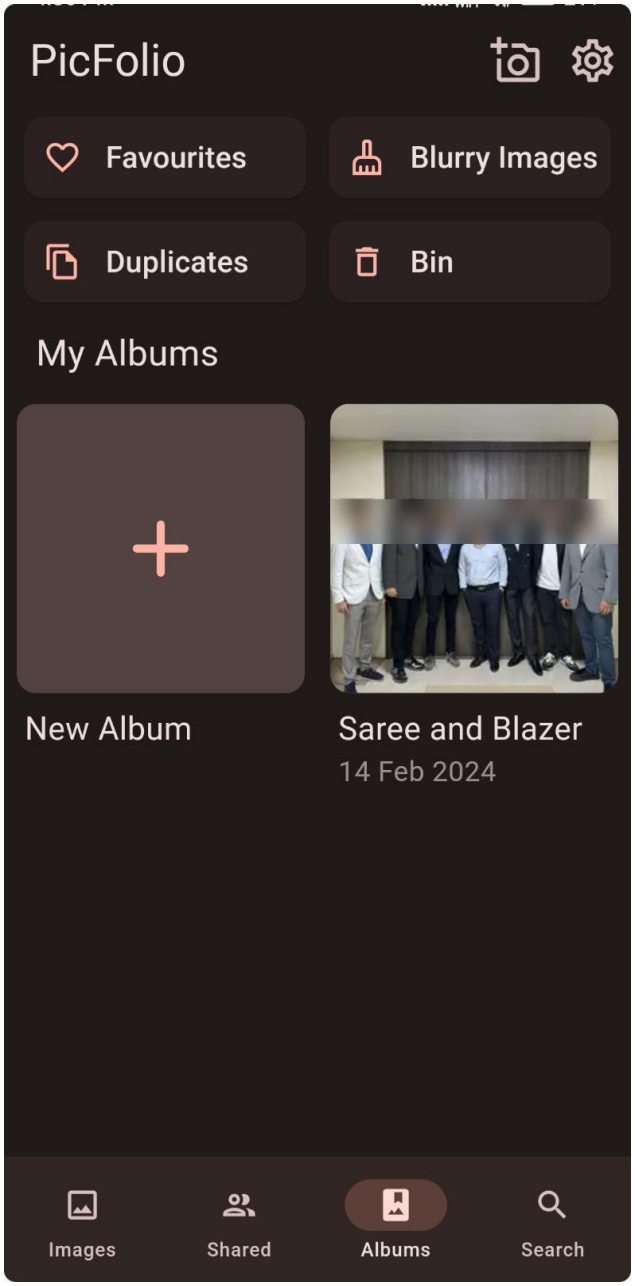


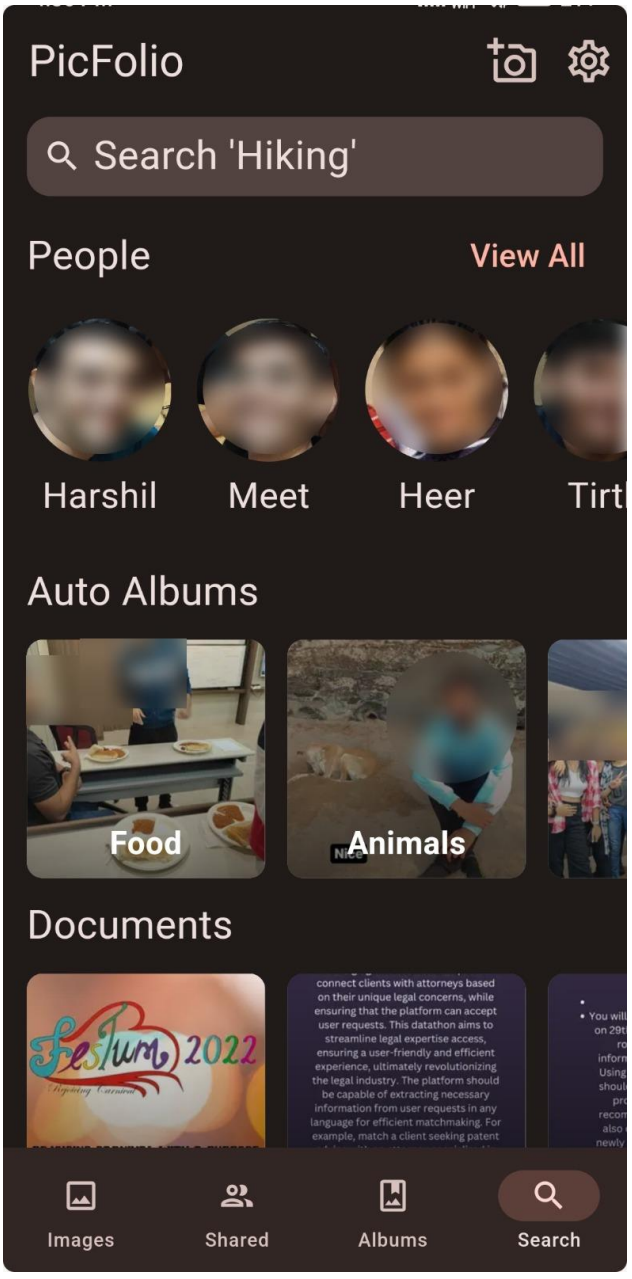

Welcome Back
Enter your credentials to login
Username
Password
Login
Forgot password?
Don't have an account? Sign Up

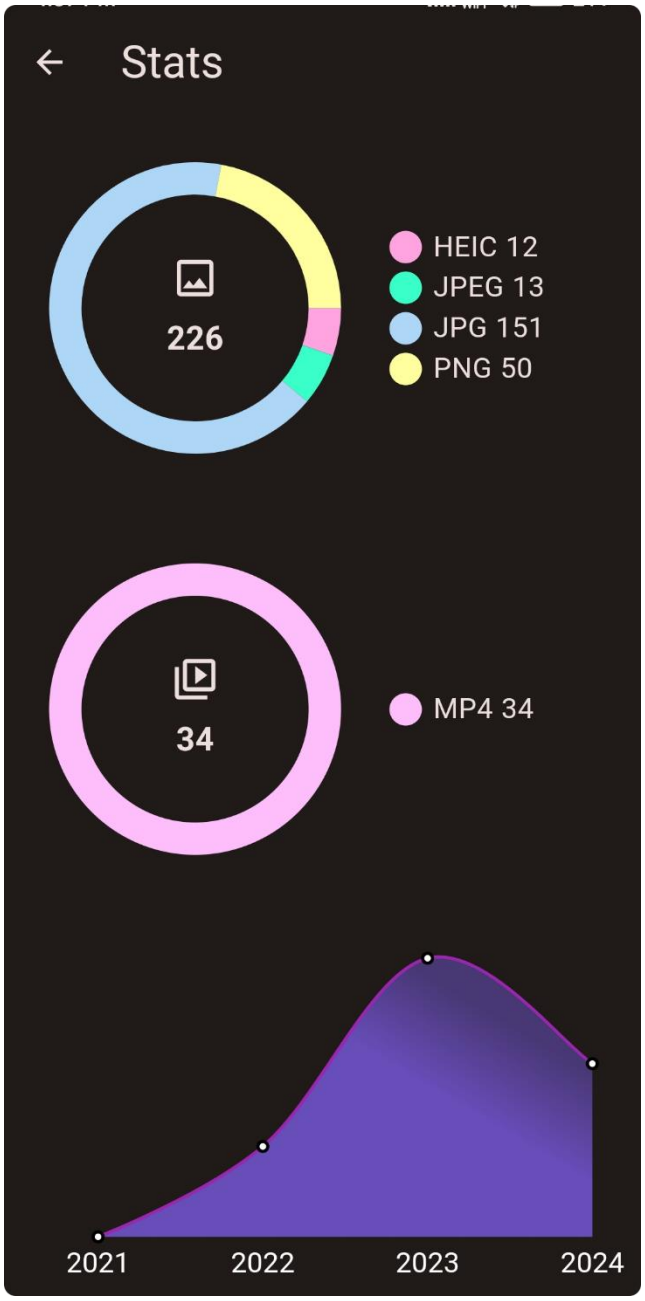

PicFolio


PicFolio
7 Apr 2024
A SUCCESSFUL ENTREPRENEUR
ENTREPRENEURSHIP DEVELOPMENT CELL
30 Mar 2024
20 Mar 2024
Khushi Tiwari
Vedant Farkade
Sai Vara Prasad Degala
6 Mar 2024
Images
Shared
Albums
Search

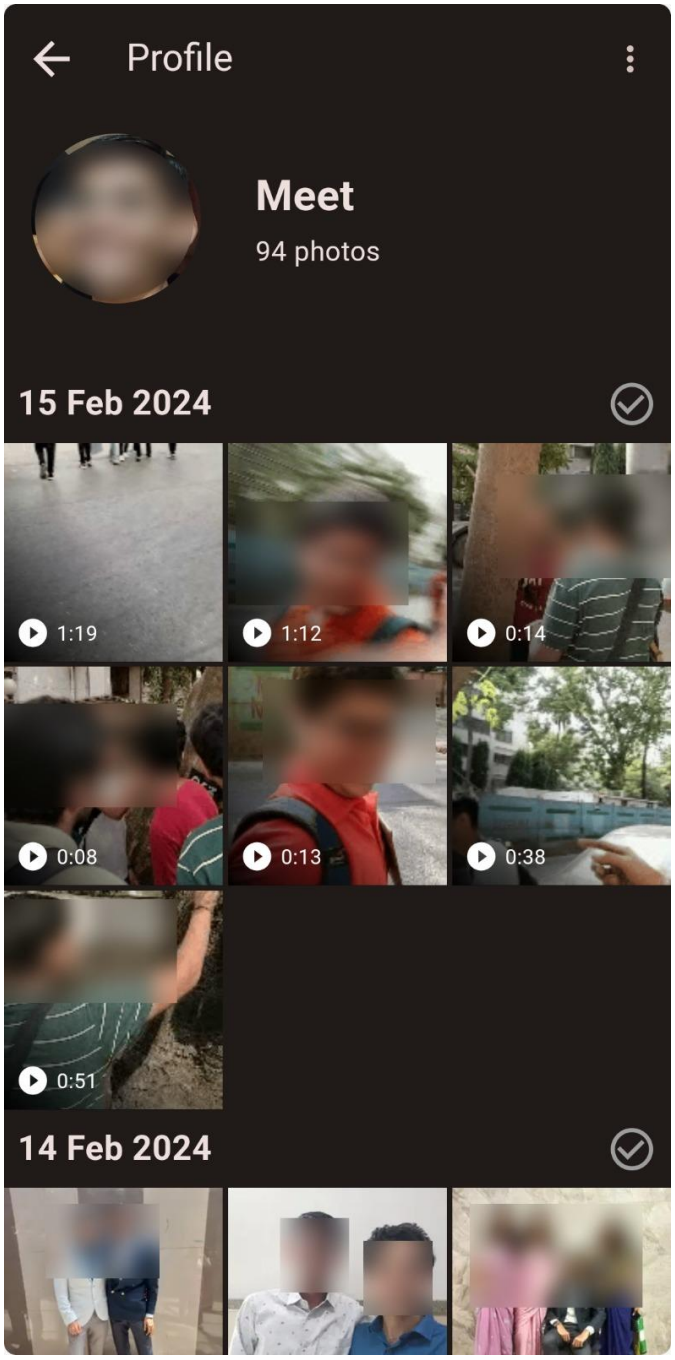

PicFolio
A
M
P
S
All users
meet244
parth28
shravo2005
5 Apr 2024
15 Feb 2024
1:19
1:12
14 Feb 2024
Images
Shared
Albums
Search

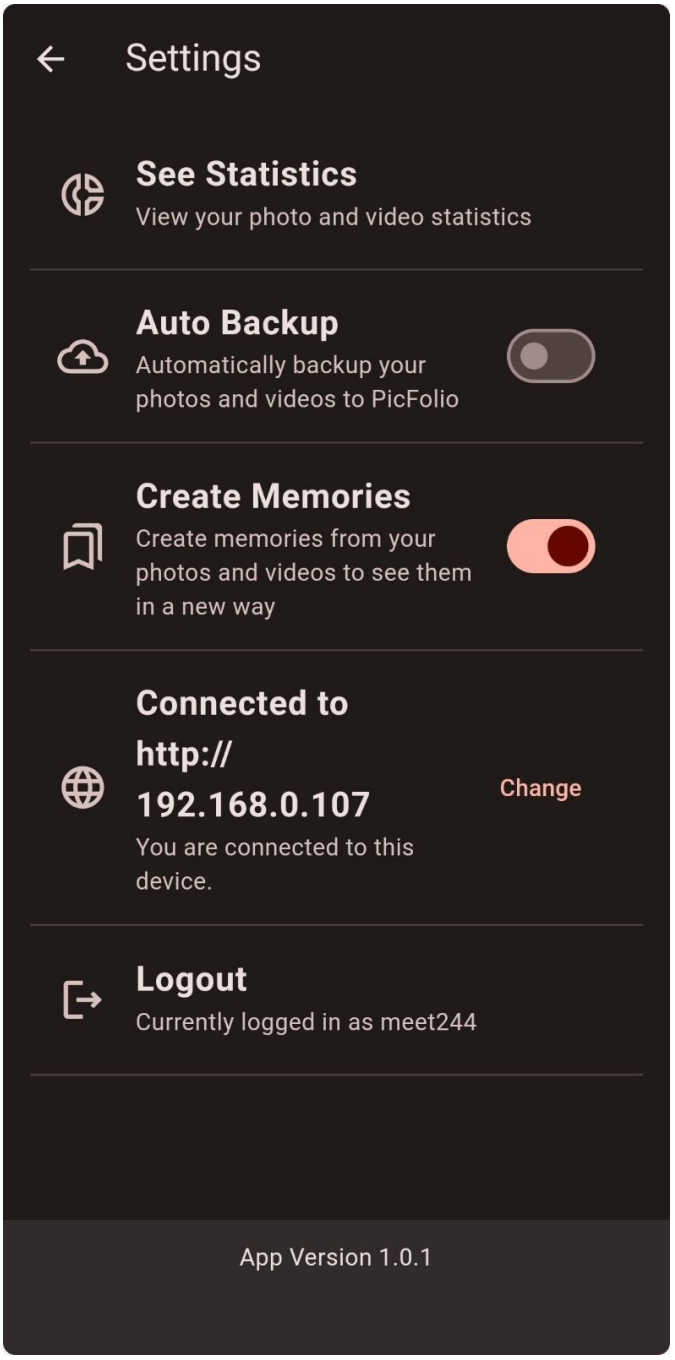

PicFolio
Favourites
Blurry Images
Duplicates
Bin
My Albums
New Album
Saree and Blazer
14 Feb 2024
Images
Shared
Albums
Search


PicFolio
Search 'Hiking'
People
View All
Harshil
Meet
Heer
Auto Albums
Food
Animals
Documents
Festum 2022
Images
Shared
Albums
Search


Stats
226
HEIC 12
JPEG 13
JPG 151
PNG 50
34
MP4 34
2021
2022
2023
2024


Duplicates
2023/10/28
2023/10/28
2023/10/28


Profile
Meet
94 photos
15 Feb 2024
1:19
1:12
0:14
0:08
0:13
0:38
0:51
14 Feb 2024


Settings
See Statistics
View your photo and video statistics
Auto Backup
Automatically backup your photos and videos to PicFolio
Create Memories
Create memories from your photos and videos to see them in a new way
Connected to http://192.168.0.107
Change
You are connected to this device.
Logout
Currently logged in as meet244
App Version 1.0.1

## 4.2 Code Snippets

Login Page for Application

```
import 'dart:convert';

import 'package:flutter/material.dart';
import 'package:photoz/color.dart';
import 'package:photoz/screens/signup.dart';
import 'package:http/http.dart' as http;
import 'package:photoz/shravani.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'package:photoz/globals.dart';

class LoginPage extends StatefulWidget {
  const LoginPage({super.key});

  @override
  _LoginPageState createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      theme: ThemeData(useMaterial3: true, colorScheme: lightColorScheme),
      darkTheme: ThemeData(useMaterial3: true, colorScheme: darkColorScheme),
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        body: Container(
          margin: const EdgeInsets.all(24),
          child: Column(
```

```
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            _header(context),
            _inputField(context),
            _forgotPassword(context),
            _signup(context),
          ],
        ),
      ),
    ),
  );
}

Widget _header(context) {
  return const Column(
    children: [
      Text(
        "Welcome Back",
        style: TextStyle(fontSize: 40, fontWeight: FontWeight.bold),
      ),
      Text("Enter your credentials to login"),
    ],
  );
}

TextEditingController passwordController = TextEditingController();
TextEditingController usernameController = TextEditingController();

Widget _inputField(context) {
  return Column(
    crossAxisAlignment: CrossAxisAlignment.stretch,
    children: [
      TextField(
        controller: usernameController,
```

```
  decoration: InputDecoration(
     hintText: "Username",
     border: OutlineInputBorder(
        borderRadius: BorderRadius.circular(18),
        borderSide: BorderSide.none),
     fillColor: Theme.of(context).inputDecorationTheme.fillColor,
     filled: true,
     prefixIcon: const Icon(Icons.person)),
),
const SizedBox(height: 20),
TextField(
  controller: passwordController,
  decoration: InputDecoration(
    hintText: "Password",
    border: OutlineInputBorder(
       borderRadius: BorderRadius.circular(18),
       borderSide: BorderSide.none),
    fillColor: Theme.of(context).inputDecorationTheme.fillColor,
    filled: true,
    prefixIcon: const Icon(Icons.password),
  ),
  obscureText: true,
),
const SizedBox(height: 20),
ElevatedButton(
  onPressed: () {
    checkLogin(usernameController.text, passwordController.text);
  },
  style: ElevatedButton.styleFrom(
    shape: const StadiumBorder(),
    padding: const EdgeInsets.symmetric(vertical: 16),
  ),
  child: const Text(
    "Login",
```

```
            style: TextStyle(fontSize: 20),
          ),
        )
      ],
    );
  }

  Widget _forgotPassword(context) {
    return TextButton(
      onPressed: () {},
      child: const Text(
        "Forgot password?",
      ),
    );
  }
```

Albums Page

```
import 'dart:convert';

import 'package:cached_network_image/cached_network_image.dart';
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;
import 'package:intl/intl.dart';
import 'package:photoz/functions/selectedImages.dart';
import 'package:photoz/globals.dart';
import 'package:photoz/screens/album.dart';
import 'package:photoz/screens/bin.dart';
import 'package:photoz/screens/duplicate.dart';
import 'package:photoz/screens/favourite.dart';
import 'package:photoz/screens/newalbum.dart';
import 'package:photoz/screens/settings.dart';
```

```
class Library extends StatefulWidget {

 const Library({Key? key}) : super(key: key);

 @override
 _LibraryState createState() => _LibraryState();
}

class _LibraryState extends State<Library> {
 List<dynamic>? albums;

 @override
 void initState() {
  super.initState();
  fetchAlbums();
 }

 Future<void> fetchAlbums() async {
  final response = await http.post(
   Uri.parse('${Globals.ip}:7251/api/list/albums'),
   body: {'username': Globals.username},
  );
  if (response.statusCode == 200) {
   // print(jsonDecode(response.body));
   setState(() {
    albums = jsonDecode(response.body);
   });
  } else {
   throw Exception('Failed to load albums');
  }
 }

 @override
 Widget build(BuildContext context) {
```

```
return Scaffold(
  appBar: AppBar(
    title: Text('PicFolio'),
    actions: <Widget>[
      IconButton(
        onPressed: () {
          // Select image from gallery
          var ret = getimage(Globals.ip, context);
          ret.then((value) {
            if (value) {
              print('Image Uploaded');
            }
          });
        },
        icon: Icon(Icons.add_a_photo_outlined, size: 32.0),
      ),
      IconButton(
        onPressed: () {
          // Open settings page
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (context) => SettingsPage(Globals.ip),
            ),
          );
        },
        icon: Icon(Icons.settings_outlined, size: 32.0),
      ),
    ],
  ),
  body: SingleChildScrollView(
    child: Center(
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
```

```
children: <Widget>[
  Row(
    mainAxisAlignment: MainAxisAlignment.center,
    children: <Widget>[
      TransparentIconButton(
          icon: Icons.favorite_outline,
          text: 'Favourites',
          onPressed: () => {
              Navigator.push(
                context,
                MaterialPageRoute(
                    builder: (context) => FavouritesScreen(
                        query: "favourite",
                        qtype: "buttons",
                      )),
              )
            }),
      TransparentIconButton(
          icon: Icons.cleaning_services_outlined,
          text: 'Blurry Images',
          onPressed: () => {
              Navigator.push(
                context,
                MaterialPageRoute(
                    builder: (context) => FavouritesScreen(
                        query: "blurry",
                        qtype: "buttons",
                      )),
              )
            }),
    ],
  ),
  Row(
    mainAxisAlignment: MainAxisAlignment.center,
```

```
  children: <Widget>[
    TransparentIconButton(
        icon: Icons.file_copy_outlined,
        text: 'Duplicates',
        onPressed: () => {
            Navigator.push(
              context,
              MaterialPageRoute(
                  builder: (context) =>
                      Duplicates(ip: Globals.ip)),
            )
          }),
    TransparentIconButton(
        icon: Icons.delete_outline,
        text: 'Bin',
        onPressed: () => {
            Navigator.push(
              context,
              MaterialPageRoute(
                  builder: (context) => BinScreen(Globals.ip)),
            )
          }),
  ],
),
Padding(
  padding:
      const EdgeInsets.symmetric(horizontal: 20, vertical: 8),
  child: Text(
    'My Albums',
    style: TextStyle(
        fontSize: 18,
        color: Theme.of(context).colorScheme.onBackground),
  ),
),
```

```
GridView.count(
  crossAxisCount: 2,
  shrinkWrap: true,
  physics: const NeverScrollableScrollPhysics(),
  childAspectRatio:
     0.85, // Adjust the value to increase or decrease vertical space
  children: [
   GestureDetector(
     onTap: () async {
      bool result = await Navigator.of(context).push(
        MaterialPageRoute(
          builder: (context) => CreateAlbum(ip: Globals.ip),
        ),
      );
      if (result == true) {
        // Handle if the return value is true
        fetchAlbums();
      }
     },
     child: LayoutBuilder(builder:
       (BuildContext context, BoxConstraints constraints) {
      return LayoutBuilder(
        builder:
          (BuildContext context, BoxConstraints constraints) {
         double containerWidth = constraints.maxWidth;
         return Column(
           children: [
            SizedBox(
              width: containerWidth,
              height: containerWidth,
              child: Container(
                margin: const EdgeInsets.all(8),
                width: double
                   .infinity, // Set the width to occupy the available space
```

```
                    decoration: BoxDecoration(
                      color: Theme.of(context)
                          .colorScheme
                          .surfaceVariant,
                      borderRadius: BorderRadius.circular(12),
                    ),
                    child: Icon(
                      Icons.add_rounded,
                      size: 60,
                      color:
                          Theme.of(context).colorScheme.primary,
                    ),
                  ),
                ),
                Padding(
                  padding:
                      const EdgeInsets.symmetric(horizontal: 8),
                  child: Align(
                    alignment: Alignment.centerLeft,
                    child: Padding(
                      padding: const EdgeInsets.symmetric(
                          horizontal: 5),
                      child: Text(
                        'New Album',
                        style: TextStyle(
                          fontSize: 16,
                          color: Theme.of(context)
                              .colorScheme
                              .onBackground,
for (int i = 0; i < (albums?.length ?? 0); i++)
  Column(
    children: [
      Container(
        margin: const EdgeInsets.all(8),
```

```
decoration: BoxDecoration(
  color: Theme.of(context).colorScheme.surfaceVariant,
  borderRadius: BorderRadius.circular(12),
),
child: LayoutBuilder(
  builder: (BuildContext context,
      BoxConstraints constraints) {
    double containerWidth = constraints.maxWidth;
    return GestureDetector(
      onTap: () {
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => Album(
                albums![i][0],
                albums![i][1],
                albums![i][3]),
          ),
        ).then((response) {
          if (response == true) {
            fetchAlbums();
          }
        });
      },
      child: ClipRRect(
          borderRadius: BorderRadius.circular(12),
          child: CachedNetworkImage(
            width: containerWidth,
            height: containerWidth,
            fit: BoxFit.cover,
            imageUrl: albums![i][2].toString().isEmpty
                ?    'https://cdn3d.iconscout.com/3d/premium/thumb/picture-
3446957-2888175.png'
                :
```

```
'${Globals.ip}:7251/api/preview/${Globals.username}/${albums![i][2].toString()}',
                    )),
              );
            },
          ),
        ),
        Padding(
          padding: const EdgeInsets.symmetric(horizontal: 8),
          child: Align(
            alignment: Alignment.centerLeft,
            child: Padding(
              padding:
                  const EdgeInsets.symmetric(horizontal: 5),
              child: Text(
                albums![i][1].toString(),
                style: TextStyle(
                  fontSize: 16,
                  color: Theme.of(context)
                      .colorScheme
                      .onBackground,
                ),
                overflow: TextOverflow.ellipsis,
              ),
            ),
          ),
        ),
        if (albums![i][3] != null)
          Padding(
            padding: const EdgeInsets.symmetric(horizontal: 8),
            child: Align(
              alignment: Alignment.centerLeft,
              child: Padding(
                padding:
                    const EdgeInsets.symmetric(horizontal: 5),
```

(0.6),

Backend user Interface code:

```
import tkinter as tk
from PIL import Image, ImageTk
import customtkinter as ctk
import threading
import ctypes
import app
import os
import socket
import qrcode
import json
from waitress import serve
from tkinter import filedialog

server_thread = None

config = None

def read_config():
    global config
    if os.path.exists('config.json'):
        with open('config.json') as f:
            config = json.load(f)
    else:
        config = {"users": ["family"], "path": ""}
        save_config()
    print("Config loaded")

def save_config():
    global config
    with open('config.json', 'w') as f:
        json.dump(config, f)
```

```
            print("Config saved")

        read_config()

        def on_stop_button_click():
            global server_thread
            if server_thread is not None and server_thread.is_alive():
                # server_thread.stop()
                thread_id = server_thread.ident
                res = ctypes.pythonapi.PyThreadState_SetAsyncExc(ctypes.c_long(thread_id),
ctypes.py_object(SystemExit))
                if res > 1:
                    ctypes.pythonapi.PyThreadState_SetAsyncExc(ctypes.c_long(thread_id), 0)
                print("Daemon thread forcefully terminated.")
                server_entry.configure(state="normal")
                server_entry.delete(0, tk.END)
                server_entry.insert(0, "0 . 0 . 0 . 0")
                server_entry.configure(state="disabled")
                port_entry.configure(state="normal")
                port_entry.delete(0, tk.END)
                port_entry.insert(0, "0000")
                port_entry.configure(state="disabled")
                toaster()
            print("Stop Photo Assistant button clicked")

        def on_start_button_click():
            global server_thread, tk_image, image_label, server_entry, port_entry, config
            # check if path is set
            if config['path'] == "":
                print("Alert")
                print("Path is not set")
                return
            if server_thread is None or not server_thread.is_alive():
                server_thread = threading.Thread(target=app.start_this, daemon=True,)
```

```
                server_thread.start()
                s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
                s.connect(("8.8.8.8", 80))
                ip = s.getsockname()[0]
                server_entry.configure(state="normal")
                server_entry.delete(0, tk.END)
                server_entry.insert(0, ip.replace('.', '   .   '))
                server_entry.configure(state="disabled")
                port_entry.configure(state="normal")
                port_entry.delete(0, tk.END)
                port_entry.insert(0, "7251")
                port_entry.configure(state="disabled")
                img = qrcode.make(f"https://picfolio.vercel.app/scan/http://{ip}")
                img.save("qr.png")
                pil_image = Image.open('qr.png')
                # crop 10 px from all edges
                much = 25
                area = (much, much, pil_image.width - much, pil_image.height - much)
                pil_image = pil_image.crop(area)
                image_label.destroy()
                resized_image = pil_image.resize((int(pil_image.width / 2.5), int(pil_image.height
/ 2.5)))
                tk_image = ImageTk.PhotoImage(resized_image)
                image_label = tk.Label(root, image=tk_image)
                image_label.place(x=70,y=150)
                s.close()
                toast()
            print("Start Photo Assistant button clicked")

        started = False
        def start_stop():
            global started
            if started:
                started = False
```

```
            on_stop_button_click()
        else:
            started = True
            on_start_button_click()

        start_button.configure(text="Stop Photo Assistant" if started else "Start Photo Assistant")
        set_data_label = ctk.CTkLabel(root, text="Set Data directory: the Images will save to this directory")
        set_data_label.grid(row=1, column=0, padx=20, pady=30)
        home_label = ctk.CTkLabel(root, text="Home Directory:")
        home_label.place(x=30, y=60)
        update_button = ctk.CTkButton(root, text="update", width=150, height=1,command=on_open_button_click,fg_color=("#FFB4A6","#AF2F1C"),hover_color=("#FFDAD4","#8D1605"),text_color=("black","white"))
        update_button.place(x=318,y=90)
        entry_var = tk.StringVar()
        entry = ctk.CTkEntry(root, textvariable=entry_var, width=260)
        entry.place(x=30,y=85)
        entry.configure(state="disabled")

        start_button = ctk.CTkButton(root, text="Start Photo Assistant", width=150, height=2, command=start_stop,fg_color=("#FFB4A6","#AF2F1C"),hover_color=("#FFDAD4","#8D1605"),text_color=("black","white"))
        start_button.place(x=320,y=220)
        # stop_button = ctk.CTkButton(root, text="Stop Photo Assistant", width=150, height=2
        # import_button.place(x=320,y=297)
        server_label = ctk.CTkLabel(root, text="Server Address:")
        server_label.place(x=320, y=340)
        server_entry_var = ctk.StringVar()
        server_entry = ctk.CTkEntry(root, textvariable=server_entry_var, width=150)
        server_entry.insert(0, "0 . 0 . 0 . 0")
        server_entry.place(x=320, y=372)
        server_entry.configure(state="disabled")
```

```
        port_label = ctk.CTkLabel(root, text="Port:")
        port_label.place(x=320, y=405)
        port_entry_var = ctk.StringVar()
        port_entry = ctk.CTkEntry(root, textvariable=port_entry_var, width=150)
        port_entry.insert(0, "0000")
        port_entry.configure(state="disabled")
        port_entry.place(x=320, y=432)
        run_checkbox_var = ctk.IntVar()
        run_checkbox = ctk.CTkCheckBox(root, variable=run_checkbox_var, text="Run PicFolio Photo Assistant on start",font=("Bahnschrift",13),text_color=('black',"white"),fg_color=("#FFB4A6","#AF2F1C"),hover_color=("#FFDAD4","#8D1605"))
        run_checkbox.place(x=22, y=437)
        #notification
        notification = ctk.CTkButton(root, text="Picfolio assistant has been started", width=250, height=1, text_color="black", fg_color="white", hover=False)
        #notification.place(x=110,y=10)
        noti = ctk.CTkButton(root, text="Picfolio assistant has been stopped", width=250, height=1, text_color="black", fg_color="white", hover=False)
        noti2 = ctk.CTkButton(root, text="Please stop the server", width=250, height=1, text_color="black", fg_color="white", hover=False)
        notified = ctk.CTkButton(root, text="Folder updated", width=250, height=1, text_color="black", fg_color="white", hover=False)


        def firstLoad():
            # set text of entery if dir is not none in config
            if config['path'] != "":
                entry.configure(state="normal")
                entry.delete(0, tk.END)
                entry.insert(0, config['path'])
                entry.configure(state="disabled")
        firstLoad()
        root.protocol('WM_DELETE_WINDOW', doSomething)  # root is your root window
root.mainloop()
```

# Chapter 5

# Conclusion and Results

## 5.1 Result Analysis

### 5.1.1 Need for Testing

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code. The increasing visibility of software as a system element and the attendant ―costs‖ associated with a software failure motivate for well-planned, thorough testing.

Instead the view of demolishing the software, the testing should be viewed as an effort to increase the software reliability. The inheritance bugs in the software are the results of human imperfection and human communications problems.

Software testing is important element of software quality assurance and represents ultimate review of specification, design and coding. A successful test is one that uncovers as yet undiscovered error. Testing demonstrates that software functions appear to be working according to the specification that performance requirements appear to have been met. Also data collected as testing is conducted provides a good indication of software reliability and quality. It is general principle of testing that all tests should be traceable to customer requirements. Also, tests should be planned long before testing begins.

### 5.1.2 Testing

**Testing of Picfolio :**

1. **Integration Testing:**

   Integration testing for Picfolio involves verifying that all components of the software, including the desktop application, mobile application, and any backend services, work together smoothly. The process begins with identifying these components and their dependencies.

   Once the components are identified, the integration strategy is determined. This strategy outlines how the components will be integrated and tested. It may include a top-down approach, where higher-level components are tested first, or a bottom-up approach, where lower-level components are tested first.

   The integration environment is then set up. This involves configuring the desktop and mobile applications to communicate with each other and any backend services. Test data is prepared, including photos with different characteristics, to test various aspects of the software.

   Integration test cases are developed to cover different scenarios, such as uploading photos from the mobile app to the desktop, performing AI and ML operations on the desktop, and verifying that the results are reflected in the mobile app.

   During test execution, the integration tests are run to ensure that all components interact correctly and produce the expected results. Error handling scenarios, such as network interruptions or invalid data, are also tested to ensure the software behaves gracefully in such situations.

2. **User Acceptance Testing:**

   Acceptance testing for Picfolio involves verifying that the software meets the requirements and expectations of its users. This type of testing is typically performed by end users or stakeholders to determine if the software is ready for release.

   For Picfolio, acceptance testing would likely involve creating test scenarios that mimic real-world usage, such as uploading photos, searching for photos, creating albums, and sharing photos with other users. Testers would interact with the software as if they were regular users, evaluating its usability, performance, and functionality.

   Feedback from acceptance testing is crucial for identifying any remaining issues or areas for improvement before the software is released to the public. It helps ensure that Picfolio

meets the needs of its users and provides a positive user experience. Acceptance testing is an essential step in the software development process, as it helps validate the software's readiness for deployment and can lead to a more successful product launch.

3. **Test Case Development:**

Test case development for Picfolio involved creating detailed scenarios to ensure that all aspects of the software were thoroughly tested. Each test case outlined a specific action or series of actions to be taken within the software, along with the expected outcome.

For example, a test case for uploading photos from the mobile app to the desktop might include steps such as selecting photos to upload, confirming the upload, and verifying that the photos appear in the correct folder on the desktop. The expected outcome would be that the photos are successfully uploaded and stored in the designated folder.

Test cases were crucial for ensuring the quality and reliability of Picfolio. They helped identify and address potential issues early in the development process, reducing the risk of bugs or errors in the final product. By thoroughly testing each aspect of the software, developers could ensure that Picfolio met the requirements and expectations of its users.

Additionally, test cases played a key role in validating the functionality and usability of Picfolio. They helped ensure that the software was intuitive and easy to use, enhancing the overall user experience.

Overall, test case development was a critical step in the development process of Picfolio. It helped ensure that the software was of high quality, met user expectations, and provided a seamless and enjoyable experience for its users.

### 5.1.3 Test cases for Picfolio

| Test Case ID | Description | Steps | Expected Result | Actual Result | Result |
|---|---|---|---|---|---|
| TC_01 | Login | 1. Enter username<br>2. Enter Password<br>3. Click login button | User should be logged in successfully | User is logged in successfully | Pass |
| TC_02 | Search | 1. Click on the search bar<br>2. Enter a keyword to search<br>3. Press enter | Relevant images should be displayed | Relevant images are displayed | Pass |
| TC_03 | High quality image load | 1. Click any high-quality image | Image should load in high quality | Image loads in high quality | Pass |
| TC_04 | Image details load | 1. Click any image<br>2. Click details button. | Image details should be displayed | Image details are displayed | Pass |
| TC_05 | Updated stats | 1. Go to “Settings”<br>2. Click on “See Statistics” | Statistics should be current and accurate | Statistics are current and accurate | Pass |
| TC_06 | New images load | 1. Scroll down the page | New images should be loaded | New images are loaded | Pass |
| TC_07 | Star high quality image from the duplicates | 1. Go to Albums tab<br>2. Click on “Duplicates” button | Image should be starred as high quality | Image is starred as high quality | Pass |
| TC_08 | QR code scanning | 1. After opening the app Scan the QR code | Relevant information should be retrieved | Relevant information is retrieved | Pass |
| TC_09 | Delete an image | 1. Click any image<br>2. Click the delete option and delete the image | Image should be deleted | Image is deleted | Pass |
| TC_10 | Favourite an image | 1. Click any image<br>2. Click the “favourite” button | Image should be marked as favourite | Image is marked as favourite | Pass |
| TC_11 | Logout of app | 1. Go to “Settings”<br>2. Click on Logout | User should be logged out | User is logged out | Pass |
| TC_12 | Change the server | 1. Go to “Settings”<br>2. Click on “change” in server settings<br>3. Change the server | App should connect to new server | App connects to new server | Pass |

## 5.2 Future Scope

PicFolio can be expanded in the future to include:

1. Integration with emerging technologies (e.g., augmented reality for photo visualization).
2. Collaboration to other 3rd party apps for features like sharing and interacting with photos in real-time.
3. Continuous improvement of AI models for more accurate photo analysis.

## 5.3 Limitations

4. Dependency on internet connectivity for certain features.
5. Compatibility issues with older hardware or software.
6. Limited Accuracy for image tagging.

## 5.4 Conclusion

In summary, the photo management application stands as a beacon of technological advancement, revolutionizing the way individuals interact with and cherish their digital memories. By seamlessly integrating across mobile and desktop platforms, it offers users unparalleled convenience in organizing, accessing, and sharing their vast photo collections. The incorporation of state-of-the-art machine learning algorithms elevates the user experience, automating tasks such as image tagging and face recognition with remarkable accuracy.

While the application's current iteration showcases immense potential, acknowledging its limitations is essential for continued growth and refinement. Addressing concerns surrounding device compatibility, internet dependency, privacy, and algorithm accuracy will be paramount in ensuring widespread adoption and sustained user satisfaction.

Looking ahead, the future of the photo management application is ripe with possibilities. By embracing cloud technology, enhancing social sharing features, and refining AI capabilities, it has the potential to become the ultimate hub for preserving and reliving life's most precious moments. Through ongoing collaboration, innovation, and dedication to user-centric design, the application is poised to shape the digital landscape, empowering individuals worldwide to curate their memories with unparalleled ease and elegance.

# References

**[1] Recognize Anything: A Strong Image Tagging Model Youcai Zhang∗1 , Xinyu Huang∗1 , Jinyu Ma∗1 , Zhaoyang Li∗1 , Zhaochuan Luo1 , Yanchun Xie1 , Yuzhuo Qin1 , Tong Luo1 , Yaqian Li1 , Shilong Liu2 , Yandong Guo3 , Lei Zhang2 1OPPO Research Institute, 2 International Digital Economy Academy (IDEA), 3AI2 Robotics**

**[2] TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems (Preliminary White Paper, November 9, 2015)**

**[3] Comparison of Face Recognition Accuracy of ArcFace, Facenet and Facenet512 Models on Deepface Framework**

**Links**

[1]Photo Prism - https://demo.photoprism.app/library/browse
[2]Ente - https://ente.io/
[3]Piwigo - https://piwigo.org/demo
[4]Immich - https://demo.immich.app/photos
[5]PiGallery2 - https://bpatrik.github.io/pigallery2/
[6]Lomorage - https://lomorage.com/

# Acknowledgement

The success of this project would not have been possible without constant advice, encouragement and support from a vast number of people.

We, the members of the team who developed “PicFolio” take immense pleasure in thanking Professor Dr. Mohd. Zafar Shaikh, Principal of Shri Bhagubhai Mafatlal Polytechnic for having permitted us to carry out this project work. We also thank our H.O.D Mr.Janardan Kulkarni for leading us the advice and helping us in our needs for making our project more successful.

We wish to express our deep sense of gratitude to our Internal Guide, Mrs. Prachi Arora for her constant guidance and useful suggestions, which helped us in completing the project work in time. Words are inadequate in offering our thanks to the entire staff of I.T/C.S.E for providing us with all amenities and facilities.

Finally, yet importantly, we would like to express our heartfelt thanks to our beloved parents for their blessings, our friends/classmates for their help and wishes for the successful completion of this project.