

# PROJECT SAIYAN AR

## Detailed Product Requirements Document (PRD)

### ***Project Idea:***

This project is inspired by the childhood imagination of transforming into a powerful anime-style character through gesture-based activation. The system allows a user to perform specific hand movements in front of a camera to trigger a realistic face transformation and cinematic energy effects in real time. The goal is to design a professional-grade real-time AR engine using Python that combines computer vision, geometric face warping, gesture recognition, and visual effects rendering.

## **1. Vision & Motivation**

The vision of Project Saiyan AR is to create a real-time Augmented Reality engine that allows users to interact with digital transformations through natural hand gestures. Unlike simple overlays or filters, this system focuses on realistic facial warping, seamless blending, lighting correction, and cinematic effects.

## **2. Problem Statement**

Existing AR filter systems are either closed ecosystems (e.g., social media platforms) or rely on heavy GPU-based deep learning pipelines that are not optimized for lightweight, modular, real-time applications. There is a need for a Python-based modular AR engine capable of gesture-driven transformation with professional-quality rendering.

### **3. Objectives**

- Implement real-time facial landmark tracking (468 landmarks).
- Implement real-time hand landmark tracking (21 landmarks per hand).
- Develop a geometric face transformation engine using triangulation.
- Design a gesture recognition state machine.
- Create cinematic particle-based energy effects.
- Maintain 30+ FPS real-time performance.

## 4. System Architecture

The system follows a modular architecture: Camera Module → Frame Processor → Face Tracker + Hand Tracker → Gesture Engine → Face Swap Engine + Effects Engine → Renderer

## 5. Technical Design

- Face Tracking: MediaPipe Face Mesh for 468 landmark detection.
- Hand Tracking: MediaPipe Hands for gesture analysis.
- Face Warping: Delaunay triangulation and affine transformation.
- Blending: Poisson seamless cloning for natural merging.
- Gesture Detection: Velocity-based and state-machine-based recognition.
- Effects Engine: Particle system, Gaussian blur glow, additive blending.

## 6. Functional Requirements

- Detect face within 100ms latency.
- Track hand movements in real time.
- Trigger transformation via swipe gesture.
- Trigger energy effect via dual-hand pose.
- Maintain stable tracking without jitter.
- Operate at minimum 30 FPS.

## 7. Non-Functional Requirements

- System latency under 150ms.
- 720p minimum resolution support.
- Modular and maintainable code structure.
- Scalable architecture for future expansion.

## 8. Development Roadmap

- Phase 1: Core Tracking Implementation.
- Phase 2: Face Swap Engine Development.
- Phase 3: Gesture Engine Integration.
- Phase 4: Effects Rendering System.
- Phase 5: Optimization and Performance Tuning.

## 9. Future Scope

Future enhancements may include AI-based real-time style transfer, multi-character transformation modes, voice-based activation, GPU acceleration using CUDA, and cross-platform deployment including web and mobile environments.