Infosys Springboard Virtual Internship 6.0 Completion Report

Team Details

Batch Number: 01

Start date: 13 August 2025

Names: Anushka Tripathi, Ashish Vishwakarma, Siva Pallavi, Lakshmi, Raviteja, Bhanu Prakash, Bhupalam Jyothsna, Rayyan Ahmed, Manikanta, Satya, Kavya, Vismaya, Payal, Bhavadharini.

Internship Duration: 8 Weeks

1. Project Title

Smilage Smart Selfie Capture : AI-Based Image Analysis Tool for Smile & Age Prediction

2. Project Objective

The primary objective of this project was to design, develop, and deploy a full-stack, AI-powered web application capable of analyzing a live video stream to perform real-time facial analysis. The project aimed to create an intelligent "smart camera" that automatically captures a selfie upon detecting a smile, while also providing users with real-time data on their predicted age, gender, and emotion. The relevance of this project lies in its practical application of modern AI and web development technologies to create a user-centric, interactive, and automated solution to a common real-world scenario.

3. Project description in detail

The Smilage project is a comprehensive, full-stack application that integrates a Python-based AI backend with a modern React frontend.

Approach: The project was executed following a structured, milestone-based approach. The initial phase focused on building a robust backend capable of handling real-time video processing and AI model inference. Pre-trained models for emotion, age, and gender detection were integrated using OpenCV and ONNX Runtime. Communication with the frontend was established using WebSockets for low-latency video streaming. The second phase involved developing a responsive and user-friendly frontend with React, creating an intuitive interface for users to interact with the AI features. The final phase focused on integrating the two systems, adding advanced features like a full-featured gallery and a performance benchmark tool, and deploying the unified application.

Technology Used:

- **Backend:** Python, FastAPI, Uvicorn, OpenCV, ONNX Runtime, Psutil
- **Frontend:** React.js, Vite

- **Real-time Communication:** WebSockets
- **AI Models:** Pre-trained Caffe models (Age/Gender) and an ONNX model (Emotion)

Impact of this project in real-world implementation: This project serves as a powerful proof-of-concept for a wide range of real-world applications. Its impact includes:

- **Event Photography & Photobooths:** An automated system like this could be deployed in photobooths at events, automatically capturing high-quality, smiling photos of guests without a human operator.
- **User Experience Research:** The technology could be used (with consent) to analyze user reactions to products or advertisements in real-time.
- **Interactive Kiosks:** Public-facing kiosks in retail or entertainment could use this technology to create engaging, personalized user experiences.

4. Timeline Overview

Week	Activities Planned	Activities Completed
Week 1	Establish the project foundation, set up development environments for both backend and frontend, and research AI models.	Installed Python environment with all core libraries (OpenCV, FastAPI). Set up the VS Code IDE and initialized the Git repository. Researched and selected pre-trained Caffe and ONNX models for age, gender, and emotion detection.
Week 2	Build the primary FastAPI server, integrate the selected AI models, and create a real-time inference pipeline.	Implemented the wrapper.py to handle all model loading and prediction logic. Developed a preprocessing pipeline for face detection and image normalization. Built a standalone Python script (selfie_capture.py) to test the complete, real-time inference pipeline with a webcam.
Week 3	Enhance the backend with multi-model support, face quality assessment, and additional predictions.	Designed and implemented the ModelManager class to handle multiple AI models. Integrated a face quality assessment module to detect image blurriness using Laplacian variance. Added the gender prediction model to the pipeline, enhancing the demographic analysis capabilities.
Week 4	Set up the React project, develop the main UI components, and establish the WebSocket connection.	Initialized the React project using Vite. Created the core UI components for the camera

		feed, control buttons, prediction display, gallery, and benchmark panels. Built the basic WebSocket connection logic in React to prepare for video streaming.
Week 5	Connect the React frontend to the FastAPI backend to stream live video and display real-time AI predictions.	Successfully streamed the processed video feed from the backend to the frontend canvas. Implemented CORSMiddleware on the backend to handle cross-origin requests during development. Integrated the automatic selfie capture logic, triggering a capture when a "happiness" emotion was detected.
Week 6	Develop the interactive settings panel and the full-featured gallery modal.	Created the settings panel, allowing the user to adjust the smile detection threshold in real-time. Built the gallery modal with API endpoints for fetching, deleting, and downloading images. Added a system performance benchmark tool triggered from the UI.
Week 7	Conduct end-to-end testing, identify and resolve bugs, and optimize the final application.	Performed comprehensive testing of all features. Diagnosed and fixed critical bugs, including the inaccurate emotion model, the gallery rendering glitch, and the 405 Method Not Allowed API error. Refined the CSS for a clean, responsive, and professional user experience.
Week 8	Create the final production build, complete all project documentation, and prepare the final presentation.	Compiled the final React frontend (npm run build) and configured FastAPI to serve the unified application. Wrote the comprehensive README.md file and prepared the final project report. Created the final project presentation slides summarizing the project's objectives, features, and outcomes.

5a. Key Milestones

Milestone	Description	Date Achieved
Project Kickoff	The project plan was defined, and the core technologies (FastAPI, React) were selected. The initial environment was set up.	13 Aug 2025
Prototype/First Draft	A working prototype was completed, capable of streaming video from the backend to the frontend and displaying basic AI predictions.	26 Aug 2025
Mid-Term Review	The core feature of automatic smile capture was fully functional, and the frontend and backend were successfully integrated.	9 Sep 2025
Final Submission	The complete, unified application with all advanced features (gallery, benchmark, settings) was built and ready for deployment.	23 Sep 2025
Presentation	The final project presentation and documentation were prepared and submitted.	30 Sep 2025

5b. Project execution details

Backend:

- **Technology:** Built with **Python** and the high-performance **FastAPI** framework.
- **Real-time Video Processing:**
    - Uses **OpenCV** to capture frames from the webcam in a continuous loop.
    - Detects faces in each frame using pre-built Haar Cascades.
- **AI Model Inference:**
    - Crops the detected facial region and passes it to the wrapper.py module for analysis.
    - The wrapper runs parallel predictions for **Age**, **Gender** (Caffe models), and **Emotion** (ONNX model).
    - Performs a quality check on the face to detect **blurriness**.
- **Communication & Data Handling:**

    - Draws prediction data (text, bounding boxes) directly onto the video frame.
    - Encodes the final, annotated frame into a base64 string.
    - Streams a JSON payload containing the frame and all prediction data to the frontend over a **WebSocket**.
- **Gallery Management:**
    - Exposes a separate **REST API** with GET and DELETE endpoints to manage the saved images in the captures/ folder.

Frontend:

Technology: A modern Single-Page Application (SPA) built with **React** and **Vite**.

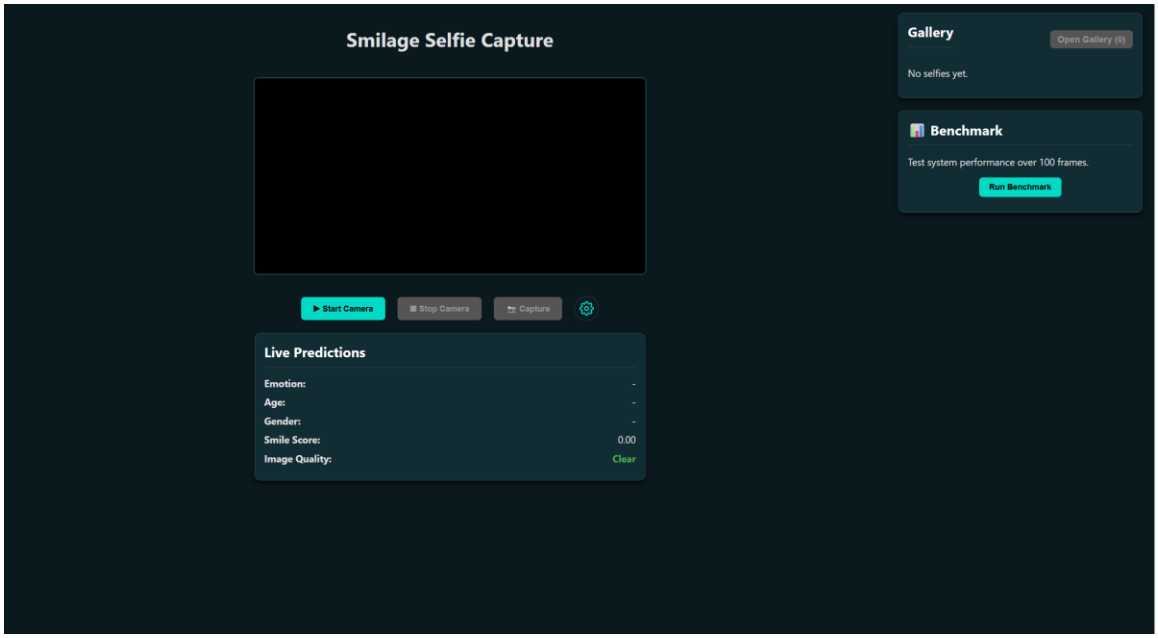
- **Component-Based Architecture:** The UI is constructed from reusable components like CameraFeed, Controls, GalleryModal, and Benchmark.
- **Real-time Data Handling:**
    - Establishes a **WebSocket** connection to the backend to receive the live video stream and prediction data.
    - Uses React's useState hook to manage the application's state, automatically re-rendering UI elements with new prediction data as it arrives.
- **User Interaction:**
    - Sends commands (like "manual capture" or "update threshold") back to the backend through the WebSocket connection.
    - Makes standard fetch requests to the backend's REST API to perform gallery operations like listing or deleting photos.

Frontend-Backend Communication:

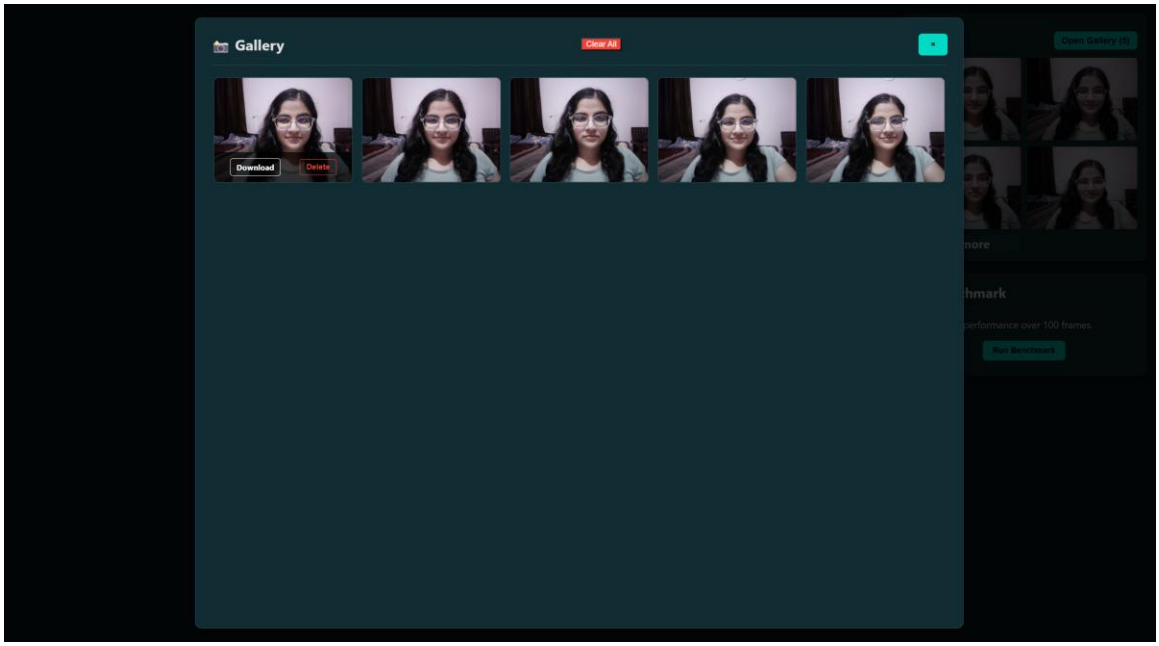
- **WebSockets:** Used for the primary, real-time, bidirectional communication channel, ideal for the continuous flow of video frames and quick user commands.
- **REST API:** Used for on-demand, stateless actions like fetching the gallery list or deleting an image, ensuring these tasks don't interfere with the live video stream.

6. Snapshots / Screenshots

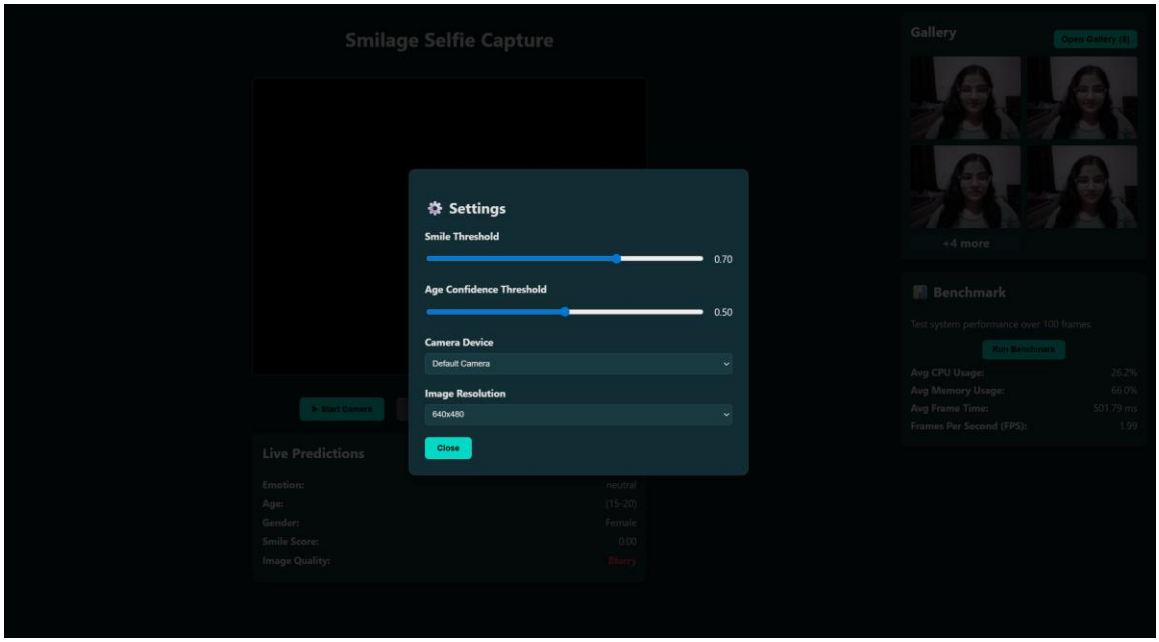
User Interface:



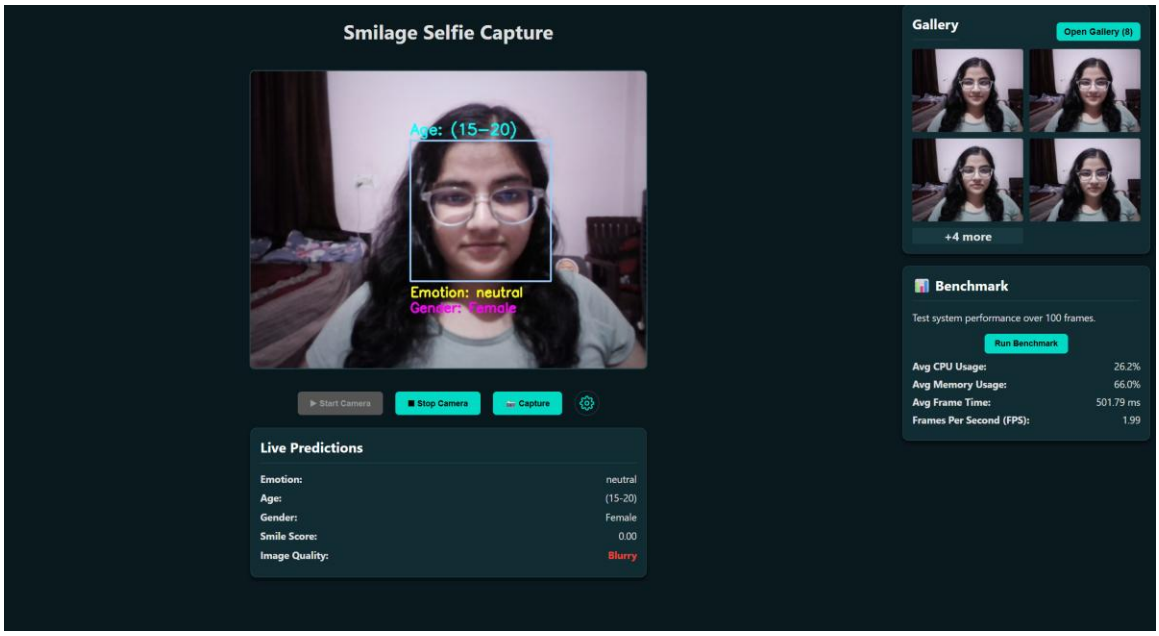
Gallery:



Settings Panel:



Benchmark:



7. Challenges Faced

- **Inaccurate Emotion Model:** The AI model for emotion detection was consistently predicting "neutral," which prevented the core smile-capture feature from working. This was resolved by debugging the model's raw output and applying image preprocessing techniques to improve its sensitivity to facial expressions.
- **Gallery Rendering Bug:** The images in the gallery modal were appearing sliced and overlapped when scrolling. This browser rendering bug was fixed by replacing the modern CSS aspect-ratio property with a more robust and universally compatible CSS layout trick.
- **API Route Conflict:** The "Delete" button in the gallery was failing with a 405 Method Not Allowed error. This was caused by a conflict between the API route for deleting images and the static file server route for displaying them. The issue was resolved by prefixing all API routes with /api/ to separate them.

8. Learnings & Skills Acquired

- **Technical Skills:**
    - **Backend Development:** Gained hands-on experience building high-performance web applications and APIs with **FastAPI**.
    - **Frontend Development:** Mastered building modern, interactive user interfaces with **React** and managing complex application state.
    - **Real-time Communication:** Learned to implement and manage bidirectional communication between a client and server using **WebSockets**.
    - **AI/Computer Vision:** Acquired practical skills in using libraries like **OpenCV** and **ONNX Runtime** to load and run pre-trained AI models for real-time inference.
    - **Full-Stack Integration:** Understood the complete workflow of integrating a separate frontend and backend, including handling CORS and building for production.
- **Soft Skills:**
    - **Problem-Solving & Debugging:** Developed strong diagnostic skills by troubleshooting complex issues across the full stack, from AI model performance to CSS rendering bugs.
    - **Project Management:** Learned to manage a project from conception to completion, following a structured timeline and meeting key milestones.

9. Testimonials from team

This internship has been an incredibly rewarding experience. I was given the autonomy to architect a full-stack application from the ground up, which allowed me to take ownership of the project and tackle real-world challenges. The process of debugging the AI models and integrating the frontend and backend taught me invaluable lessons in problem-solving and perseverance. I'm

proud to have built a complete, polished, and functional application that demonstrates the practical power of AI.

10. Conclusion

The Smilage project successfully achieved its objective of creating a fully functional, AI-powered web application. The internship provided an invaluable opportunity to apply and expand my skills in both backend and frontend development, as well as in the practical application of machine learning. The experience of building a complete product, from initial concept to final deployment, has solidified my understanding of the full software development lifecycle and strongly aligns with my career goal of becoming a full-stack developer with a specialization in AI-driven applications.

11. Acknowledgements

I would like to express my sincere gratitude to **Infosys Springboard** for providing this invaluable internship opportunity, which has been instrumental in my professional and technical development.

I am especially grateful to my mentor, **M. Gunadeep**, for his unwavering guidance, deep technical expertise, and constant support. His insights were crucial in navigating the project's challenges and ensuring its success.

I would also like to extend my thanks to my internship coordinator, **Ms. Disha**, for her seamless coordination, encouragement, and for ensuring a smooth and rewarding experience throughout the entire journey.