

C E U N I V E R S



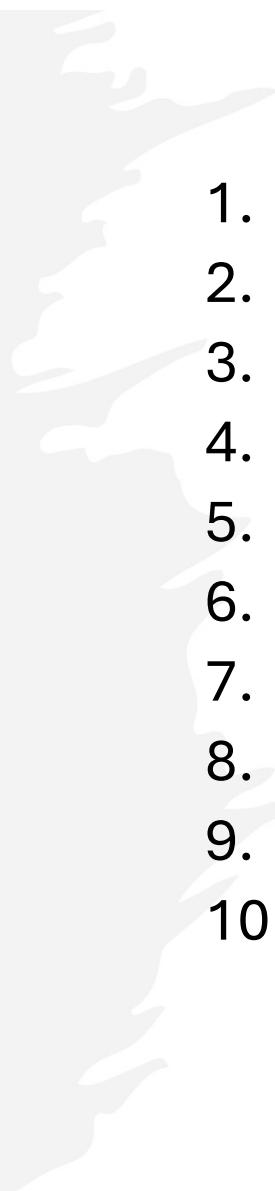
FaceFind

**AI-Powered Event Photo
Discovery**

CS 691

Team 7

AGENDA

- 
1. Team Member Roles & Responsibilities
 2. Problem Statement & Hypothesis
 3. Project Description
 4. Personas
 5. Technologies
 6. Algorithms
 7. Project Schedule & Cadence
 8. Team Working Agreement
 9. Sprint 0 Retrospective
 10. Group Wiki Page Link

Team Member Roles and Responsibilities

Jash Berawala: Full Stack Lead

- Backend API (Node.js/Express)
- Database Architecture (MongoDB)
- ML Integration
- API Development & Testing

Jay Shah: Frontend Developer

- React.js Application
- Admin Dashboard Interface
- Guest Photo Gallery
- Responsive Design & State Mgmt

Niraj Patil: ML Engineer

- Face Detection & Recognition
- Python Microservice (FastAPI)
- Face Clustering (DBSCAN)
- Model Performance Optimization

Chetan Patel: Database Administrator

- AWS S3 Integration
- CI/CD Pipeline (GitHub Actions)
- Docker Containerization
- Security & QR Code System

Problem Statement

Current Challenges in Event Photography

Manual Photo Distribution

Photographers spend 5-6 hours per event manually sorting and sharing photos

Guest Frustration

Attendees must scroll through hundreds/thousands of photos to find themselves

Delayed Access

Photos delivered days/weeks after events, missing social media engagement window

Privacy & Access Control

No easy way to give specific guests access while maintaining event privacy

Project Description

What is FaceFind?

- ✓ Intelligent web platform automating event photo distribution
- ✓ Admin Dashboard: Photographers manage events & upload photos
- ✓ Guest Interface: Attendees discover photos instantly via selfie
- ✓ AI Engine: Deep learning face detection & clustering
- ✓ Smart Matching: Cosine similarity for <5 sec results
- ✓ Built on MERN stack + Python ML microservice

How It Works

1. Create event → Generate QR
2. Upload photos → AI processes
3. Guest scans QR → Takes selfie
4. System matches faces
5. Personalized gallery delivered

Persona 1 — The Photographer : “Alex the Memory-Maker”

Background:

Full-time photographer shooting 3-4 events/week. Spends 5-6 hours manually organizing and sharing photos per event.

Goals:

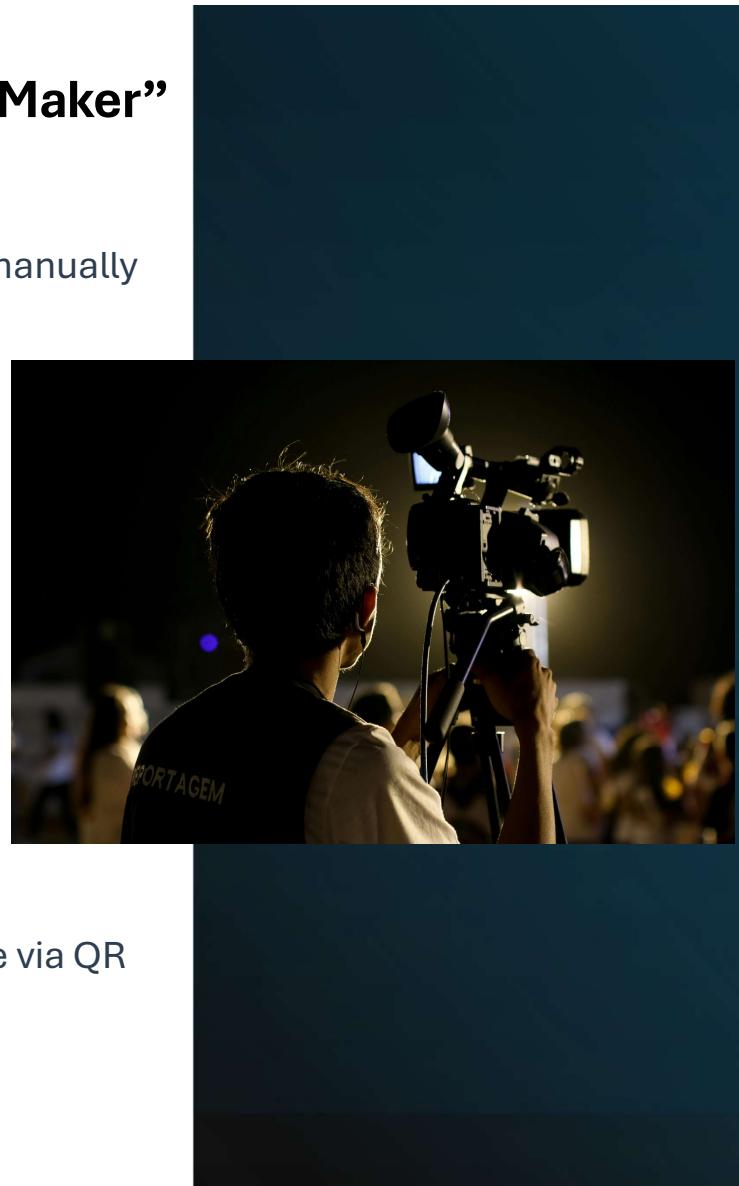
- Reduce photo delivery time from hours to minutes
- Improve client satisfaction with faster turnaround
- Focus more on photography and client acquisition

Pain Points:

- Manual folder creation and link sharing
- Constant client requests: 'When will photos be ready?'
- No easy way to provide partial access

How FaceFind Helps:

Creates event in 30 seconds, AI handles distribution, guests self-serve via QR code, frees 5+ hours per event





Persona 2: Wedding Guest - Marcus Thompson

Background:

28-year-old marketing manager, attends 8-10 weddings/year. Frustrated by delayed photo access.

Goals:

- Get photos immediately while memories are fresh
- Easily find photos of himself without scrolling
- Share to social media during the event

Pain Points:

- Receives photo links 3-4 weeks after events
- Must scroll through 800+ photos to find himself
- Misses social media engagement window



Persona 3 — The Event Organizer: “Jordan the Coordinator”

Background:

Senior Events Manager organizing 15-20 company events annually. Needs secure, measurable photo distribution.

Goals:

- Quick, secure photo delivery to employees
- Track engagement and measure ROI

How FaceFind Helps:

Private events with email whitelist, detailed analytics, 73% engagement in 48hrs

Technologies

Frontend

- React.js
- Redux Toolkit
- Tailwind CSS
- Socket.io Client

Backend

- Node.js + Express
- MongoDB + Redis
- JWT Auth
- Socket.io

Machine Learning

- Python + PyTorch
- FastAPI
- InsightFace
- OpenCV + NumPy

Cloud & DevOps

- AWS S3 + CloudFront CDN
- Docker Containers
- GitHub Actions (CI/CD)
- Nginx Reverse Proxy

Complete MERN Stack

MongoDB + Express + React + Node.js
+ Python ML Microservice
+ Cloud Infrastructure (AWS)

Algorithms



**Machine Learning /
AI Algorithms Used**



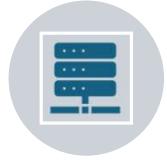
**Face Detection:
MTCNN /
RetinaFace**



**Face Embedding:
ArcFace
(InsightFace)**



**Similarity Search:
FAISS / pgvector**



**Clustering
(Future): K-Means /
DBSCAN**



**Pattern
Recognition: Deep
Learning CNNs**



Project Schedule & Cadence

- **Sprint Cadence**
- **Sprint Length: 2 weeks**
- **Daily Standup: 15 minutes**
- **Sprint Planning: Start of each sprint**
- **Sprint Review: End of each sprint**
- **Retrospective: End of each sprint**
- **Weekly Tech Sync: Twice per week**

Team Working Agreement



This agreement outlines the expectations, responsibilities, and commitments of all team members throughout the duration of the project. By signing this document, each member agrees to uphold the standards listed below to ensure a productive, respectful, and collaborative working environment.

1. Communication

- The team will use **Slack / Microsoft Teams/ Google Meet** as the primary communication platform.
- All members agree to respond to messages **within 12 hours**.
- Important updates, blockers, or delays must be communicated promptly and transparently.

2. Meetings

- All team members agree to:
- Attend **all sprint ceremonies**, including:
 - Sprint Planning
 - Daily Standups

Team Working Agreement



3. Work Expectations

- Each team member commits to:
- Completing assigned tasks before the deadline.
- Avoiding last-minute submissions that create unnecessary pressure on others.
- Submitting work for peer review before merging into the main branch.
- Communicating blockers early so the team can assist and adjust accordingly.

4. Accountability

- If a team member cannot complete a task, they must notify the team immediately.
- The team agrees to maintain a supportive, no-blame culture.
- Each member is responsible not only for their own tasks but also for contributing to the overall success of the sprint.

5. Team Values

- We agree to uphold the following values:
- Respect for each other's time, effort, and contributions
- Honest and open communication
- Collaboration and willingness to help teammates
- Professionalism, positivity, and shared ownership of outcomes

Retrospective

What Went Well

Clear understanding of project vision

Roles assigned effectively

Strong alignment on architecture

Good communication setup

What Can Be Improved

Need more clarity on data flow diagrams

Improve time estimation accuracy

Actions for Next Sprint

Create detailed architecture diagrams

Build initial FastAPI + Next.js skeleton

These become Sprint 1 backlog items

Group Wiki Page Link

<https://github.com/JashBerawala/Pace-Project-Team7/wiki>



Thank you

