Drone delivery has the potential to improve medical access, reduce delays, and support patients in remote areas. However, the benefits of such a system can only be realized if it is secure. Medical deliveries involve controlled substances, private patient data, and time-critical treatments. If medication reaches the wrong person, the consequences can include compromised treatment outcomes, theft of pharmaceutical goods, and potential health risks.

Before my contribution, the system had no mechanism for confirming who is placing an order or who is collecting it. Anyone with access to the interface could request a delivery, and drones had no way to verify that the person who approaches them is the legitimate recipient. My additions try to directly solve this problem by creating a secure chain of identity verification from the moment a delivery is requested to the moment it is picked up.

The primary stakeholders are:

- Dispatch staff coordinating medical deliveries
- Medical personnel who authorize medication distribution
- Patients receiving medicines

By incorporating OTP verification and face recognition, the system ensures that only authorized users can request deliveries and only the intended recipients can unlock and collect them. This establishes trust, minimizes the risk of fraud, and aligns the system with real-world security expectations.

OTP systems are widely used in banking, email verification, and high-security platforms due to their strengths over static passwords. An OTP is:

- Randomly generated
- Single-use
- Valid only for a short time window

This drastically reduces the risk of credential theft, replay attacks, or brute-force access, as an attacker would need to guess a 6-digit number before it expires. With one million possible combinations, the probability of a correct random guess is extremely low, and the short expiry window further minimizes risk.

OTP allows the system to confirm that the delivery requester truly has access to the registered email address associated with that account. It is a lightweight yet effective authentication method.

Face recognition has become a mainstream authentication mechanism in mobile devices, payment systems, airport gates, and secure workplaces because it is:

- Fast and contactless
- Harder to fake than passwords
- Based on unique biological traits

Biometrics represent “something you are”, and when combined with OTP (“something you have”), they form a multi-factor authentication system, significantly raising security.

I implemented face recognition using InsightFace, a leading open-source facial analysis library based on deep learning. InsightFace produces facial embeddings, high-dimensional vectors encoding facial features which can then be compared using cosine similarity. The model is known for high accuracy under controlled conditions, often achieving over 90% recognition accuracy according to published benchmarks.

Together, OTP and face recognition create an effective end-to-end verification pipeline, enhancing safety for medical delivery scenarios.

OTP Service Implementation

The OTP service was developed in Java using Spring Boot:

- Each OTP is generated using SecureRandom, producing a cryptographically safe 6-digit code.
- OTPs are stored in an in-memory concurrent hash map keyed by email, with timestamps and a 5-minute expiry.
- JavaMailSender sends the OTP email to users, meaning the system verifies both identity and email ownership.
- After successful verification or expiration, OTPs are deleted, preventing replay attacks.

In a full-scale implementation, this service can be extended to include rate limiting, persistent user profiles, lockout mechanisms after repeated failures, or integration with authenticator apps.

Face Recognition Service Implementation

The facial recognition component is built using Python and FastAPI:

- InsightFace's "antelopev2" model is used for face detection and embedding extraction.
- The system takes two images: a stored reference image and a live capture from a webcam or drone camera.
- It extracts embeddings for each face and compares them using cosine similarity.
- If the similarity score exceeds a threshold (0.45 in my prototype), the system identifies a match.
- FastAPI is lightweight, fast, and ideal for ML-backed APIs

The prototype currently captures images via webcam, but integrating this with a drone's onboard camera is straightforward. The system can easily be expanded to accept uploaded images or integrate more advanced anti-spoofing techniques, such as liveness detection.

The system is built with modularity in mind, the OTP verification and facial recognition operate as independent services, each with a clear responsibility. The delivery logic interacts with these services through API calls, following clean separation-of-concerns. Using Python for the ML component and Java for the backend allows each part of the system to use the most appropriate ecosystem for its required tasks.