This is the master technical documentation for the **EMsync: Emergency Coordination Hub** project as of December 22, 2025 12:30 PM It details the architecture, logic, and implementation of all completed phases, with a specific focus on the ongoing development of Phase 3.

EMsync Master Project Documentation

1. Project Core Objective

The primary goal of EMsync is to develop an integrated platform that shifts emergency medical response from reactive to **predictive and proactive** by synchronizing ambulances and hospitals in real-time[111111111].

---

2. Phase 1: Emergency Severity & Resource Prediction

- **Goal:** To utilize AI/ML models to predict patient severity and forecast critical ICU resource needs before arrival[222222222].
- **Dataset:** Processed **MIMIC-IV-ED** (~425,000 ED visits) and **MIMIC-IV-ICU** datasets[3333333333333333].
- **Model:** Successfully trained **7 distinct XGBoost models**[444444444].
- **Performance:**
  - **Severity Prediction:** 71% Balanced Accuracy (Categories: Critical, Moderate, Low Urgency)[555555555555].
  - **Resource Forecasting:** 65–77% Balanced Accuracy for needs like Ventilators, Vasopressors, Dialysis, Sedatives, and Opioids[666666666].
- **Current State:** Integrated into a Streamlit-based simulation dashboard for real-time inference[777777777].

---

3. Phase 2: Smart Ambulance Routing

- **Goal:** To implement routing that dynamically adapts to real-time traffic to minimize transit delays[888888888].
- **Architecture:** Client-server model with a **Flask (Python) backend** and **React frontend**[999999999].
- **Key Integration: OpenRouteService (ORS) API** for dynamic pathfinding based on live road data[10101010101010101010101010].
- **Functionality:** Smooth client-side animation for live tracking and secure API key management via environment variables[1111111111111111].

---

4. Phase 3: Inter-Hospital Transfer Coordination (Ongoing)

- **Goal:** To establish a secure system for real-time resource visibility, bed booking, and standardized digital handoffs[121212121212121212].

4.1. Implementation Strategy: Virtual Hospital Network

Since real-world hospital API access is not available for development, we have implemented a **Virtual Registry** logic. Each hospital is a database entry with dynamic resource counts (ICU beds, specialists)[13].

4.2. Command & Environment Log

Bash

```
# Environment Creation
conda create --name virtual_network_emsync_env python=3.9
conda activate virtual_network_emsync_env

# Dependency Installation
pip install Flask-SQLAlchemy pycryptodome flask-socketio flask-cors
```

4.3. Database Logic (hosital_model.py)

Logic: A relational schema to track virtual hospital nodes and transfer session states.

Python

```
from flask_sqlalchemy import SQLAlchemy

db = SQLAlchemy()

class Hospital(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(100), nullable=False)
    lat = db.Column(db.Float, nullable=False)
    lon = db.Column(db.Float, nullable=False)
    total_icu_beds = db.Column(db.Integer, default=10)
    available_icu_beds = db.Column(db.Integer, default=5)
    has_ventilator = db.Column(db.Boolean, default=True)
    has_specialist = db.Column(db.Boolean, default=True)

class TransferSession(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    transfer_id_encrypted = db.Column(db.String(255), unique=True, nullable=False)
    origin_hospital_id = db.Column(db.Integer, db.ForeignKey('hospital.id'))
    destination_hospital_id = db.Column(db.Integer, db.ForeignKey('hospital.id'))
    status = db.Column(db.String(50), default='PENDING') # PENDING -> ACCEPTED -> EN_ROUTE
-> COMPLETED
    sbar_json = db.Column(db.Text, nullable=True)
```

4.4. Security Logic (security.py)

Theory: AES-256 Symmetric Encryption protects Patient Identifiable Information (PII) during transfer[14141414141414141414].

Python

```
import base64
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
from Crypto.Random import get_random_bytes

SECRET_KEY = b'EMsync_Secret_Key_2025_Phase_3!!'

def encrypt_patient_data(data_string):
    iv = get_random_bytes(16)
    cipher = AES.new(SECRET_KEY, AES.MODE_CBC, iv)
    ct_bytes = cipher.encrypt(pad(data_string.encode('utf-8'), AES.block_size))
    return base64.b64encode(iv + ct_bytes).decode('utf-8')

def decrypt_patient_data(encrypted_id):
    raw_data = base64.b64decode(encrypted_id)
    iv, ct = raw_data[:16], raw_data[16:]
    cipher = AES.new(SECRET_KEY, AES.MODE_CBC, iv)
    return unpad(cipher.decrypt(ct), AES.block_size).decode('utf-8')
```

4.5. Clinical Logic (sbar_logic.py)

Logic: Automatically maps data from Phase 1 (Severity) and Phase 2 (Vitals) into the **SBAR** (Situation, Background, Assessment, Recommendation) medical standard[1515151515151515].

Python

```
import json

def generate_sbar_report(patient_info, vitals, ml_prediction, resource_needs):
    sbar_report = {
        "S": {"urgency": ml_prediction, "complaint": patient_info.get("primary_complaint")},
        "B": {"age": patient_info.get("age"), "history": patient_info.get("history")},
        "A": {"vitals": vitals, "predicted_needs": resource_needs},
        "R": {"suggested_unit": "ICU" if ml_prediction == "Critical" else "Ward"}
```

```
    }
    return json.dumps(sbar_report)
```

4.6. Backend API Logic (backend_api_app.py)

Logic: The central hub managing resource allocation and secure handshakes.

- **GET /api/hospitals**: Provides real-time visibility of the virtual network.
- **POST /api/request_transfer**: Atomically reserves a bed, encrypts patient PII, and generates an SBAR report.

```
Python
```

```
# [Full verified code previously provided, including both GET and POST endpoints]
```

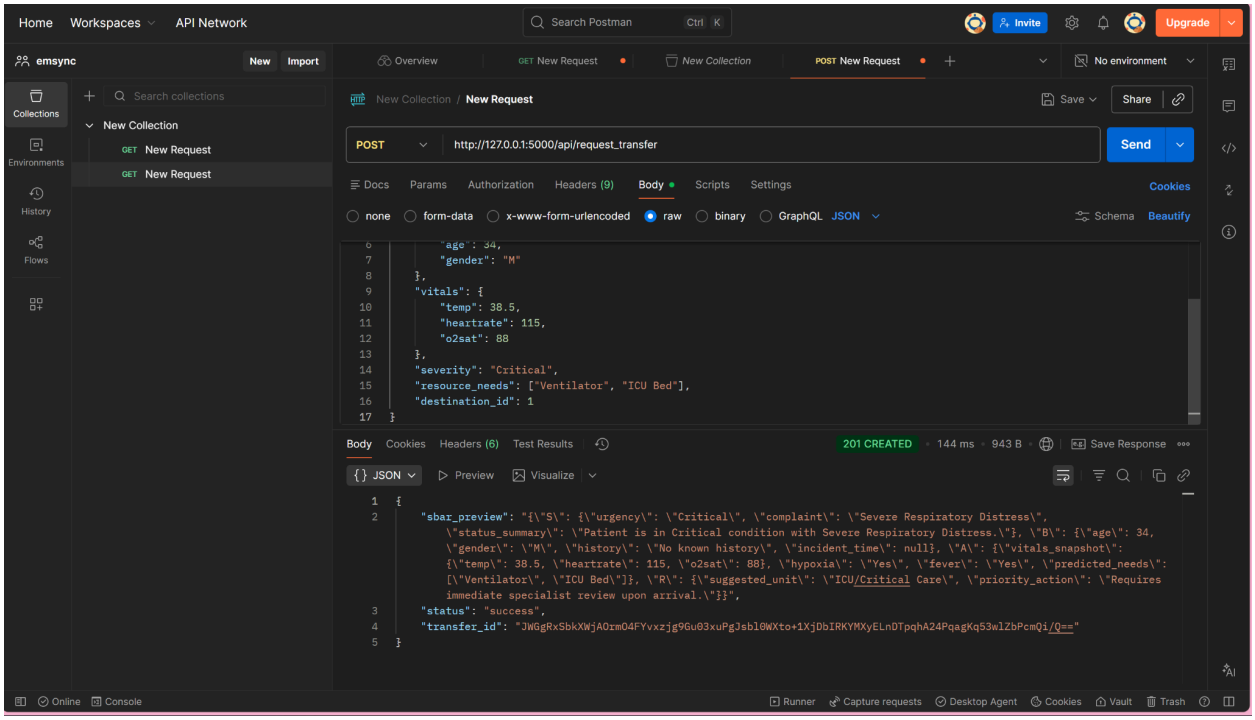
---

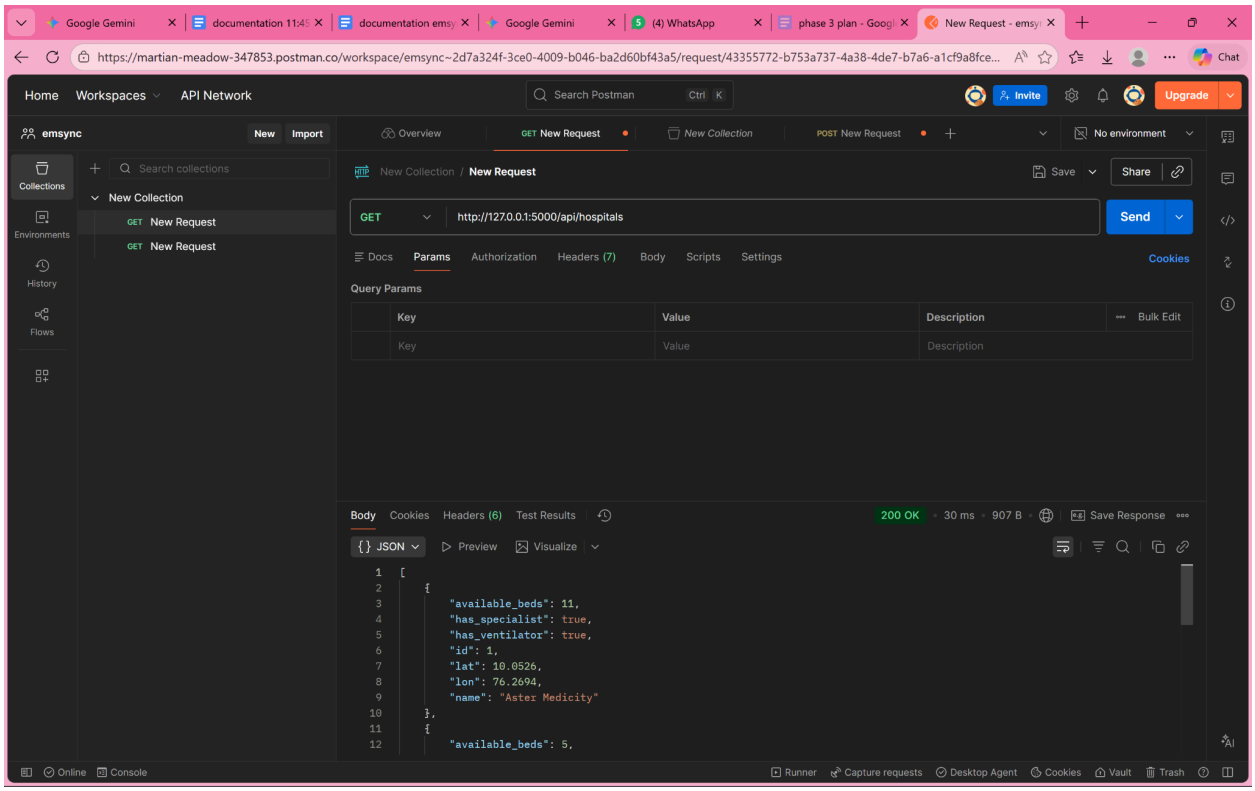
5. Project Status Summary

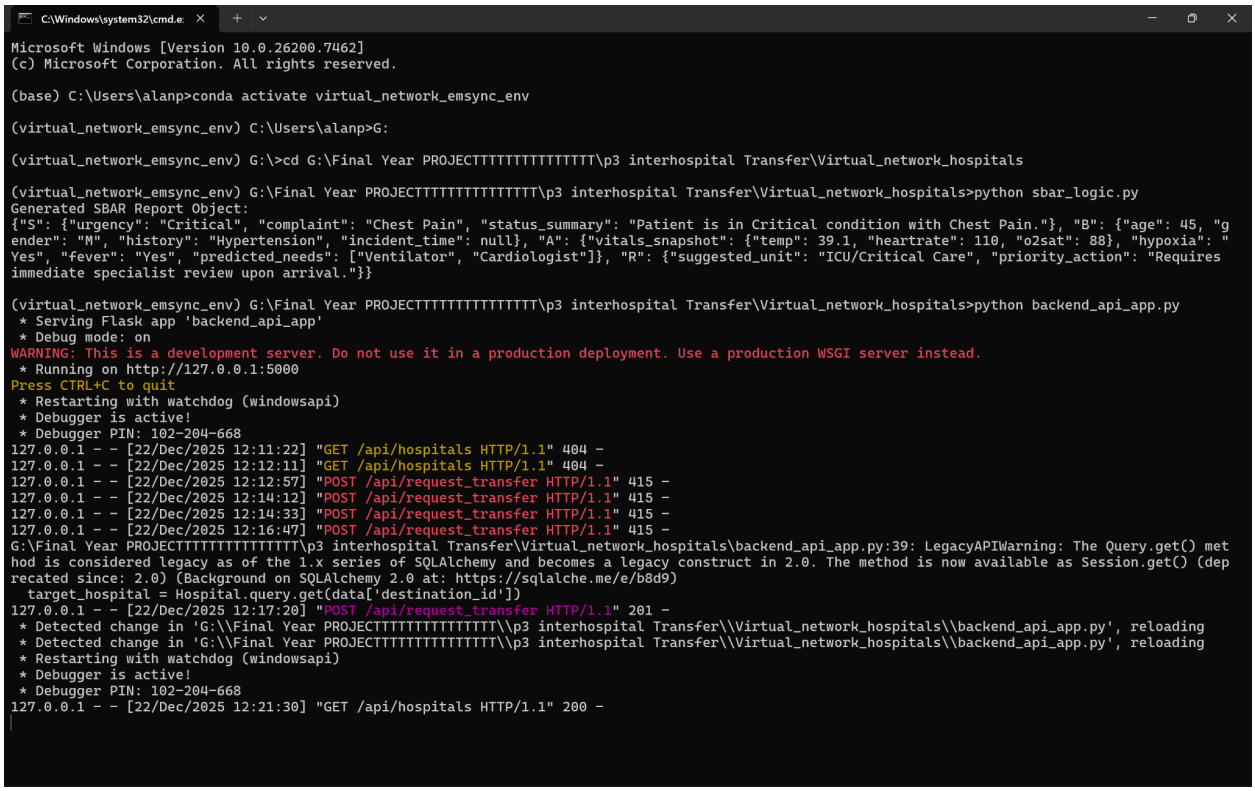
Milestone	Status
Phase 1: ML Models	**Completed** (71% Severity Acc) [16161616]
Phase 2: Routing	**Completed** (ORS API + React) [17171717]
Phase 3: Coordination	**Ongoing** (Backend Logic Verified) [18]
Database Seeding	**Completed** (Aster, Lourdes, Rajagiri hospitals)
Encryption Engine	**Completed** (AES-256) [19191919]

Clinical Handoff	**Completed** (Automated SBAR) 20202020

End of Documentation. Ready to proceed with **Step 5: Real-Time Synchronization via WebSockets**.




Google Gemini
documentation 11:45
documentation emsy
Google Gemini
(4) WhatsApp
phase 3 plan - Googl
New Request - emsy
https://martian-meadow-347853.postman.co/workspace/emsync~2d7a324f-3ce0-4009-b046-ba2d60bf43a5/request/43355772-b753a737-4a38-4de7-b7a6-a1cf9a8fce...
Chat
Home
Workspaces
API Network
Search Postman
Ctrl K
Invite
Upgrade
emsync
New
Import
Overview
GET New Request
New Collection
POST New Request
No environment
Collections
Environments
History
Flows
Search collections
New Collection
GET New Request
GET New Request
New Collection / New Request
Save
Share
GET
http://127.0.0.1:5000/api/hospitals
Send
Docs
Params
Authorization
Headers (7)
Body
Scripts
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Key
Value
Description
Body
Cookies
Headers (6)
Test Results
200 OK
30 ms
907 B
Save Response
{ } JSON
Preview
Visualize
1 [
2 {
3 "available_beds": 11,
4 "has_specialist": true,
5 "has_ventilator": true,
6 "id": 1,
7 "lat": 10.0526,
8 "lon": 76.2694,
9 "name": "Aster Medicity"
10 },
11 {
12 "available_beds": 5,
Online
Console
Runner
Capture requests
Desktop Agent
Cookies
Vault
Trash


C:\Windows\system32\cmd.e
Microsoft Windows [Version 10.0.26200.7462]
(c) Microsoft Corporation. All rights reserved.
(base) C:\Users\alanp>conda activate virtual_network_emsync_env
(virtual_network_emsync_env) C:\Users\alanp>G:
(virtual_network_emsync_env) G:\>cd G:\Final Year PROJECTTTTTTTTTTTTTTT\p3 interhospital Transfer\Virtual_network_hospitals
(virtual_network_emsync_env) G:\Final Year PROJECTTTTTTTTTTTTTTT\p3 interhospital Transfer\Virtual_network_hospitals>python sbar_logic.py
Generated SBAR Report Object:
{"S": {"urgency": "Critical", "complaint": "Chest Pain", "status_summary": "Patient is in Critical condition with Chest Pain."}, "B": {"age": 45, "gender": "M", "history": "Hypertension", "incident_time": null}, "A": {"vitals_snapshot": {"temp": 39.1, "heartrate": 110, "o2sat": 88}, "hypoxia": "Yes", "fever": "Yes", "predicted_needs": ["Ventilator", "Cardiologist"]}, "R": {"suggested_unit": "ICU/Critical Care", "priority_action": "Requires immediate specialist review upon arrival."}}
(virtual_network_emsync_env) G:\Final Year PROJECTTTTTTTTTTTTTTT\p3 interhospital Transfer\Virtual_network_hospitals>python backend_api_app.py
* Serving Flask app 'backend_api_app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with watchdog (windowsapi)
* Debugger is active!
* Debugger PIN: 102-204-668
127.0.0.1 - - [22/Dec/2025 12:11:22] "GET /api/hospitals HTTP/1.1" 404 -
127.0.0.1 - - [22/Dec/2025 12:12:11] "GET /api/hospitals HTTP/1.1" 404 -
127.0.0.1 - - [22/Dec/2025 12:12:57] "POST /api/request_transfer HTTP/1.1" 415 -
127.0.0.1 - - [22/Dec/2025 12:14:12] "POST /api/request_transfer HTTP/1.1" 415 -
127.0.0.1 - - [22/Dec/2025 12:14:33] "POST /api/request_transfer HTTP/1.1" 415 -
127.0.0.1 - - [22/Dec/2025 12:16:47] "POST /api/request_transfer HTTP/1.1" 415 -
G:\Final Year PROJECTTTTTTTTTTTTTTT\p3 interhospital Transfer\Virtual_network_hospitals\backend_api_app.py:39: LegacyAPIWarning: The Query.get() method is considered legacy as of the 1.x series of SQLAlchemy and becomes a legacy construct in 2.0. The method is now available as Session.get() (deprecated since: 2.0) (Background on SQLAlchemy 2.0 at: https://sqlalche.me/e/b8d9)
target_hospital = Hospital.query.get(data['destination_id'])
127.0.0.1 - - [22/Dec/2025 12:17:20] "POST /api/request_transfer HTTP/1.1" 201 -
* Detected change in 'G:\\Final Year PROJECTTTTTTTTTTTTTTT\\p3 interhospital Transfer\\Virtual_network_hospitals\\backend_api_app.py', reloading
* Detected change in 'G:\\Final Year PROJECTTTTTTTTTTTTTTT\\p3 interhospital Transfer\\Virtual_network_hospitals\\backend_api_app.py', reloading
* Restarting with watchdog (windowsapi)
* Debugger is active!
* Debugger PIN: 102-204-668
127.0.0.1 - - [22/Dec/2025 12:21:30] "GET /api/hospitals HTTP/1.1" 200 -