Project Phases Template

Project Title:

Hematovision : Advanced Blood Cell

Classification UsingTransfer Learning

# Team ID : LTVIP2025TMID38900

# Phase-1: Brainstorming & Ideation

1. Technological Innovation Concepts

a. Real-Time Blood Imaging

* A wearable or implantable device that continuously analyzes blood composition (e.g., glucose, oxygen, toxins)
* Could use nanotechnology or spectroscopic sensors
* Integrated with a smart device to alert abnormalities

b. AR/VR Visualization of Blood Flow

* Augmented reality tools for doctors to “see” inside the patient’s vascular system in real-time
* Used in surgeries or diagnostics
* Gamified training for medical students

c. AI Blood Analysis System

* An AI model that takes input from a blood test and produces:
  + Visual diagnostic maps (e.g., heatmaps showing abnormalities)
  + Predictive insights (e.g., risk scoring for diseases)
  + Personalized treatment suggestions

2. Medical Applications

a. Hematological Disease Tracking

* Visualization dashboards for conditions like leukemia, hemophilia, sickle cell anemia
* Tracks treatment progress via blood analytics
* Alerts patients and doctors in case of dangerous thresholds

b. Remote Diagnostics

* Use of micro blood samplers and image-based diagnostics
* Data transmitted to cloud for visualization and remote analysis

3. Consumer Health Tools

* Hematovision Ring/Band: Like a smart ring that monitors blood characteristics (e.g., hemoglobin, hydration, etc.)
* Diet & Supplement Advisor: AI visualizes micronutrient levels and suggests personalized plans
* Fitness Mode: Tracks lactate, hydration, and blood oxygen during workouts

4. Research & Educational Tools

* 3D simulation of blood reactions (e.g., clotting, oxygen transport) for schools and universities
* An open-source platform where researchers visualize blood data and model new theories

5. Art & Science Hybrid

* “Blood as Data” art installation: Using real-time blood data to drive visuals, sound, or interactive displays
* Great for museums or public health awareness events

💡 Ideation Techniques You Can Use

* SCAMPER: Substitute, Combine, Adapt, Modify, Put to another use, Eliminate, Reverse
* Mind Mapping: Place “Hematovision” in the center, branch to tech, medical, education, consumer, ethics
* SWOT Analysis: Analyze strengths, weaknesses, opportunities, threats of Hematovision as a product/system

🧬 Future Possibilities

* Integration with genomic data: View how genes affect blood markers
* Use with brain-machine interfaces: Control devices based on blood-oxygen levels or neurotransmitter indicators
* Application in emergency triage: Visual prioritization based on internal bleeding detection

# Phase-2: Requirement Analysis

1. 📌 Problem Statement

There is a lack of real-time, visual, and accessible tools for comprehensive blood analysis and monitoring. Existing blood tests are:

* Invasive
* Periodic (not continuous)
* Dependent on lab infrastructure
* Difficult to interpret visually by non-specialists

Hematovision aims to solve this by providing real-time, intuitive visualization and analytics of blood health using AI, imaging, sensors, and/or cloud technologies.

2. 🎯 Objectives

* Enable real-time or rapid blood analysis
* Provide clear, intuitive visualizations for professionals and/or consumers
* Detect and alert users about abnormalities (e.g., anemia, infections, clotting issues)
* Support integration with existing medical records or health platforms
* Potentially allow remote monitoring and diagnostics

3. 👥 Target Users

| User Group | Needs |
| --- | --- |
| Doctors / Clinicians | Fast diagnostics, clear visual reports, alerts for conditions |
| Patients / Consumers | Easy-to-understand visuals, personalized health insights |
| Hospitals / Labs | Integration with systems (EHRs), automation of reports |
| Medical Students / Educators | Educational simulations of blood behavior or diseases |
| Remote Health Providers | Telehealth-compatible diagnostics and dashboards |

This phase focused on identifying and documenting the technical, functional, and resource requirements necessary to successfully build and deploy the waste classification system.

4. 🔍 Functional Requirements

| ID | Requirement | Description |
| --- | --- | --- |
| FR1 | Blood Sample Interface | Device or method to capture and input blood data |
| FR2 | Real-Time Analysis Engine | Analyze blood for key markers (e.g., RBC count, hemoglobin, glucose) |
| FR3 | Visualization Module | Generate visual representations (graphs, AR views, simulations) |
| FR4 | Alert System | Notify users of dangerous readings (threshold-based) |
| FR5 | Data History / Tracking | Store and visualize trends over time |
| FR6 | Integration with EHR | Export to or sync with electronic health records |
| FR7 | Multi-user Access | Roles for doctors, patients, admins |
| FR8 | Report Generation | Create shareable medical reports (PDF, email, API) |

5. ⚙️ Non-Functional Requirements

| Category | Description |
| --- | --- |
| Performance | Fast response time (e.g., < 5s for visual rendering) |
| Scalability | Handle multiple patients/data streams |
| Security | HIPAA/GDPR compliant, encrypted data storage and transfer |
| Reliability | 99.9% uptime, redundant systems |
| Usability | Intuitive UI for both professionals and laypeople |
| Portability | Web, mobile, tablet, and wearable-compatible |
| Interoperability | Works with lab equipment and medical databases |

6. 🛑 Constraints

* Regulatory approvals (FDA, CE, etc.)
* Hardware limitations (e.g., sensor accuracy)
* Cost and accessibility for developing regions
* Data privacy and ethical concerns
* Accuracy requirements for medical-grade diagnostics

7. 📈 Success Criteria

* System accuracy > 95% compared to traditional lab tests
* User satisfaction score > 80%
* Clinician adoption in hospitals or practices
* Regulatory clearance achieved (if needed)
* Reduction in diagnostic time by X%

# Phase-3: Project Design

In this phase, the focus was on designing both the architecture of the AI model and the structure of the web application. The goal was to ensure that the solution was modular, scalable, and easy to maintain.

## 1  Frontend (User Interface):

## Web (React/Angular)

## Mobile App (Flutter/React Native)

## Optional AR/VR integration (Unity, WebXR)

##  API Gateway:

## Manages client requests

## Load balancing, authentication, and rate limiting

##  Application Layer (Backend):

## Business logic

## Blood analysis workflows

## Report generation

##  AI/ML Engine:

## Models trained for blood pattern recognition

## Anomaly detection, risk prediction

##  Sensor/Device Input Module:

## Integrates with external hardware (e.g., micro blood analyzers, smart wearables)

## Accepts lab results, live sensor data, image uploads

##  Data Layer:

## Relational DB (PostgreSQL) for structured records

## NoSQL DB (MongoDB) for logs, visual datasets

## Cloud storage for images, AR/VR models

## 

# Phase-4: Project Planning (Agile Methodologies)

1. 📌 Project Overview

Project Name: Hematovision  
Approach: Agile (Scrum-based)  
Sprint Duration: 2 weeks  
Project Timeline (MVP): ~12–16 weeks for MVP  
Team Roles:

| Role | Responsibility |
| --- | --- |
| Product Owner | Defines features, prioritizes backlog |
| Scrum Master | Facilitates Agile process, removes blockers |
| Dev Team | Engineers (frontend, backend, ML, IoT) |
| UX/UI Designer | Designs user flows and interfaces |
| QA Engineer | Testing strategy and execution |
| Medical Advisor | Validates clinical requirements |

2. 🔁 Agile Iterative Phases

🧭 Phase 1: Sprint 0 – Planning & Setup (1–2 weeks)

* Define MVP scope
* Set up dev environments and repos
* Initial UI wireframes
* Backlog grooming
* Technology stack finalization

🧱 Phase 2: Core Sprints (6–8 sprints, 2 weeks each)

| Sprint | Key Deliverables |
| --- | --- |
| Sprint 1 | User authentication, roles, UI scaffolding |
| Sprint 2 | Data ingestion from mock blood inputs |
| Sprint 3 | Core processing engine + test data pipeline |
| Sprint 4 | AI/ML models (e.g., hemoglobin, WBC detection) |
| Sprint 5 | Visualization module (charts, blood flow UI) |
| Sprint 6 | Alert & notification system |
| Sprint 7 | Report generation + EHR integration |
| Sprint 8 | End-to-end user flow testing, UI polishing |

🔚 Phase 3: Finalization & Release (2–3 weeks)

* Full QA + UAT testing
* Compliance & security audits
* Deploy to staging, then production
* Prepare launch documentation
* Stakeholder demo

3. 📝 Product Backlog Examples

| Epic | User Story | Priority |
| --- | --- | --- |
| Authentication | As a user, I want to securely log in | High |
| Blood Input | As a clinician, I want to upload blood test data | High |
| Visualization | As a user, I want to view trends in blood health | Medium |
| AI/ML Analysis | As a doctor, I want to get predictive insights | High |
| Alerts | As a patient, I want to be alerted if levels are abnormal | High |
| Reports | As a user, I want to download/share medical reports | Medium |
| Device Integration | As an admin, I want to connect with a wearable | Low (post-MVP) |

4. 🧮 Agile Estimation Techniques

* Story points (1 = trivial, 5 = complex)
* Planning poker for team estimation
* Burndown chart tracking sprint velocity

5. 📅 Sprint Cadence Example

| Day | Activity |
| --- | --- |
| Day 1 | Sprint Planning (define sprint goal, assign tasks) |
| Day 2–10 | Daily Stand-ups, Dev work, QA, Reviews |
| Day 10 | Sprint Review & Demo |
| Day 11 | Sprint Retrospective |
| Day 12 | Sprint Planning (next sprint) |

6. 📊 Agile Tools

* Project Management: Jira / Trello / ClickUp
* Version Control: GitHub / GitLab
* CI/CD: GitHub Actions, CircleCI
* Design: Figma / Adobe XD
* Communication: Slack, Notion, Zoom
* Testing: Postman, Selenium, Jest, Pytest

7. ✅ Definition of Done (DoD)

* Feature complete and unit tested
* Code reviewed and merged
* Integrated with main system
* UI/UX approved
* Deployed to staging
* Documentation written

8. 🧪 QA & Testing in Agile

* Continuous integration testing
* Functional testing during sprints
* Regression testing before releases
* Automated test suite developed alongside code
* End-user testing in final sprints

# Phase-5: Project Development

🔁 Development Cycle Overview (Agile Execution)

Each sprint (2 weeks) includes:

1. Sprint Planning – Define backlog items for the sprint
2. Daily Standups – Track blockers and progress
3. Development Tasks – Backend, frontend, ML, integrations
4. Code Reviews – Pull requests and quality checks
5. Sprint Demo – Working features demonstrated to stakeholders
6. Retrospective – Feedback loop for continuous improvement

🧱 Development Breakdown by Layers

1. 🔙 Backend Development

Tech Stack: Python (FastAPI or Flask), Node.js (if JS-based), PostgreSQL, MongoDB

| Feature | Tasks |
| --- | --- |
| API Design | RESTful endpoints for auth, data input, visualization |
| Data Storage | Schema for users, blood data, predictions, reports |
| Processing Engine | Middleware to clean, normalize, and store blood data |
| Report Generation | Create PDF/HTML reports with charts & summaries |
| Security | JWT auth, RBAC, encryption at rest and in transit |

2. 🧠 AI/ML Development

Tech Stack: Python, TensorFlow / PyTorch, Scikit-learn, OpenCV

| Feature | Tasks |
| --- | --- |
| Model Training | Classification for blood markers (e.g., anemia) |
| Image Recognition | Detect WBC/RBC counts from microscope images |
| Predictive Analytics | Risk scoring and anomaly detection |
| API Integration | Serve predictions via REST endpoints |
| Model Validation | Accuracy, precision, recall evaluations |

3. 🎨 Frontend Development

Tech Stack: React / Vue.js (Web), Flutter / React Native (Mobile)

| Feature | Tasks |
| --- | --- |
| Dashboard | View live blood data, trends, and predictions |
| Visualization | Charts, graphs, and (optional) 3D/AR blood flow |
| Alerts UI | Notifications for critical markers |
| Upload & Input | Data input interface (manual or device) |
| User Profile | Settings, access roles, export tools |

4. 📲 Device / IoT Integration (Optional, Future Phase)

Tech Stack: BLE, Microcontroller SDKs (e.g., Arduino, Raspberry Pi)

| Feature | Tasks |
| --- | --- |
| Device Sync | Connect blood sensor or wearable device |
| Data Streaming | Live data sent to cloud/backend |
| Calibration | Sync sensor output with ML expectations |
| Failover | Handle disconnects or signal loss gracefully |

5. 🔒 Security Implementation

| Layer | Security Feature |
| --- | --- |
| API | OAuth 2.0, JWT tokens |
| Storage | AES-256 encryption |
| Transport | HTTPS with TLS 1.3 |
| Access Control | Role-based permissions for doctor/patient/admin |
| Compliance | HIPAA/GDPR logging and data handling policies |

CODE:

A screen shot of a computer program

AI-generated content may be incorrect.

# Phase-6: Functional & Performance Testing

1. Functional Testing

Goal: Ensure that every feature works as expected based on the business requirements.

📋 Key Functional Test Areas

| Feature | Test Cases |
| --- | --- |
| User Authentication | Login, logout, password reset, role-based access |
| Blood Data Input | Manual input, sensor data upload, API submission |
| Data Validation | Correct formats, error handling, missing/invalid data |
| Visualization | Charts render correctly, data updates in real-time |
| ML Predictions | Outputs are generated and interpreted correctly |
| Alert System | Threshold breaches trigger accurate notifications |
| Reports | Export generates correct, formatted summaries |
| Role Access Control | Patient vs Doctor vs Admin permissions |

🧪 Functional Testing Types

| Type | Purpose | Tools |
| --- | --- | --- |
| Unit Testing | Test individual components | Pytest, Jest, Mocha |
| Integration Testing | Verify data flows between modules | Postman, Insomnia |
| End-to-End Testing | Simulate user journey | Selenium, Cypress |
| API Testing | Test REST endpoints | Postman, SuperTest |
| UI Testing | Validate UI interactions | Cypress, Playwright |

⚙️ 2. Performance Testing

Goal: Ensure the system performs under expected (and peak) loads with acceptable latency, throughput, and scalability.

📈 Key Performance Metrics

| Metric | Target (MVP) |
| --- | --- |
| API Response Time | < 300ms (normal load) |
| Dashboard Load Time | < 1.5s |
| Concurrent Users | ≥ 1,000 |
| Data Upload Throughput | ≥ 50 requests/sec |
| Alert Delivery Latency | < 1s |

🧪 Performance Testing Types

| Test Type | Description | Tools |
| --- | --- | --- |
| Load Testing | Simulate normal user volume | Locust, JMeter |
| Stress Testing | Push system beyond limits | k6, BlazeMeter |
| Spike Testing | Test response to sudden surges | k6 |
| Endurance Testing | Run for long periods (e.g., 24h) | JMeter, Artillery |
| Scalability Testing | Measure how well system scales | AWS auto-scaling, Kubernetes testing |

OUTPUT:

A screenshot of a medical test

AI-generated content may be incorrect.

A close-up of a microscope

AI-generated content may be incorrect.

THANK YOU