**Work Plan and Instructions for Product Development Team - StrikeO Case Management Solution**

**Work Breakdown Structure (WBS)**

**Phase 1: Planning and Initial Setup (Weeks 1-2)**

**1.1 Product Vision and Feature Prioritization**

* **Objective**: Clearly define the features and scope of the product.
* **Task**:
  + Collaborate with CTG’s stakeholders to finalize product requirements.
  + Outline key features (Real-time monitoring, AI alerts, KYC onboarding, reporting modules, semantic search, and case management).
  + Prioritize the features based on importance and project timeline.
* **Deliverable**: Product roadmap and feature prioritization document.
* **Resources**: CTO, Product Owner, Business Analyst.

**1.2 Define Technical Architecture**

* **Objective**: Design a scalable, secure, and efficient system architecture.
* **Task**:
  + Define the architecture to support fiat and crypto transactions, real-time monitoring, AI-based fraud detection, and semantic search capabilities.
  + Decide on the technology stack (detailed below).
  + Set up data flow diagrams, integration points with APIs (e.g., Elliptic, Fireblocks, etc.), and blockchain interfaces.
* **Deliverable**: Detailed architecture diagram and data flow chart.
* **Resources**: CTO, Lead Architect, Backend Developers.

**Phase 2: UX/UI Design and Wireframes (Weeks 2-4)**

**2.1 User Journey Mapping**

* **Objective**: Understand the needs of compliance officers, admins, and other key users.
* **Task**:
  + Design user journeys for different roles such as compliance officers, admins, investigators, and data analysts.
  + Map out how users will interact with the system for onboarding, KYC verification, case management, and alerts.
* **Deliverable**: User journey maps for different roles.
* **Resources**: UX/UI Designers, Product Owner, Compliance Team.

**2.2 Wireframe Creation**

* **Objective**: Create a visual representation of the platform's layout.
* **Task**:
  + Design wireframes for each module (Dashboard, Case Management, Reporting, Alerts, Transaction Monitoring, and User Management).
  + Prototype the user interface to ensure easy navigation and compliance with UX/UI standards.
* **Deliverable**: Wireframes for key screens and user flow diagrams.
* **Resources**: UX/UI Designers, Product Owner.

**Phase 3: Backend Development - Core Modules (Weeks 4-12)**

**3.1 Backend Setup and API Development**

* **Objective**: Set up core backend infrastructure to support real-time monitoring, case management, and integration with AI models.
* **Task**:
  + Set up the server environment using **Python (Django)** for API development.
  + Integrate **PostgreSQL** for secure data storage of user profiles, transaction logs, and compliance records.
  + Implement **Node.js** for real-time features such as alerts and transaction streaming.
* **Deliverable**: Backend infrastructure set up, API endpoints for data ingestion, and transaction handling.
* **Resources**: Backend Developers, DevOps Engineer.

**3.2 AI Model Integration (AML/CTF Monitoring)**

* **Objective**: Develop and integrate AI models for transaction monitoring and fraud detection.
* **Task**:
  + Set up **TensorFlow** or **PyTorch** for AI/ML model development.
  + Build AI models to detect suspicious transaction patterns, such as layering, smurfing, or rapid movement between crypto wallets.
  + Integrate **Elliptic**/**Chainalysis/Crystal Blockchain APIs** for cryptocurrency monitoring and risk scoring.
* **Deliverable**: AI models for AML/CTF monitoring integrated into the backend.
* **Resources**: AI Engineers, Data Scientists, Backend Developers.

**3.3 Transaction Monitoring (Fiat & Crypto)**

* **Objective**: Build modules to monitor both fiat and cryptocurrency transactions.
* **Task**:
  + Develop modules for tracking fiat transactions, using data feeds from traditional financial systems (bank accounts, SWIFT, etc.).
  + Build cryptocurrency monitoring by integrating with APIs (Chainalysis, Fireblocks, etc.) to monitor blockchain transactions.
  + Implement risk models (RAG) and flagging systems for suspicious activities.
* **Deliverable**: Fiat and cryptocurrency transaction monitoring modules.
* **Resources**: Backend Developers, Blockchain Developers, AI Engineers.

**Phase 4: Frontend Development - Dashboards & User Interface (Weeks 6-14)**

**4.1 Dashboard and Reporting Module**

* **Objective**: Develop real-time dashboards for compliance officers to manage cases, monitor transactions, and view alerts.
* **Task**:
  + Use **React.js** and **D3.js** for building interactive dashboards displaying transaction flows, alerts, and case statuses.
  + Implement the reporting module for generating **IFTI**, **SMR**, **TTR**, and other compliance reports.
  + Design **admin panels** for managing users, roles, and access permissions.
* **Deliverable**: Functional real-time dashboard and reporting module.
* **Resources**: Frontend Developers, UX/UI Designers.

**4.2 Semantic Search Implementation**

* **Objective**: Build a semantic search engine to allow for contextual searching within the platform.
* **Task**:
  + Integrate **ElasticSearch** for full-text search functionality across case files, transaction logs, and reports.
  + Use **NLP models** to enhance search capabilities, enabling compliance officers to find relevant data quickly using natural language queries.
* **Deliverable**: Fully functional semantic search engine integrated into the UI.
* **Resources**: Frontend Developers, Data Scientists, Backend Developers.

**Phase 5: Case Management and AI Alerts (Weeks 10-16)**

**5.1 Case Management System**

* **Objective**: Develop the case management system for compliance officers to track, escalate, and close cases.
* **Task**:
  + Build case creation, assignment, and tracking functionalities, enabling officers to manage the lifecycle of compliance investigations.
  + Ensure each case includes transaction details, user profiles, AI-generated risk scores, and audit trails.
* **Deliverable**: Case management system with complete workflows and alert integration.
* **Resources**: Backend Developers, Frontend Developers.

**5.2 AI-Powered Alerts & Escalation System**

* **Objective**: Develop AI-powered alerts that detect suspicious activities in real time.
* **Task**:
  + Build an **alerting system** where AI models flag suspicious transactions and trigger real-time alerts for compliance officers.
  + Implement an **escalation workflow**, allowing for cases to be escalated based on the severity (RAG rating) or volume of suspicious activities.
* **Deliverable**: AI-powered alert system and escalation workflow.
* **Resources**: AI Engineers, Backend Developers.

**Phase 6: Security, Testing, and Deployment (Weeks 16-20)**

**6.1 Security Integration**

* **Objective**: Ensure the platform is secure and compliant with regulatory standards.
* **Task**:
  + Implement **OpenSSL** for encryption of sensitive data (transactions, user profiles, case files).
  + Use **Vault by HashiCorp** for secure storage of secrets and API keys.
  + Implement **OpenID Connect/OAuth2.0** for secure user authentication and role-based access control (RBAC).
* **Deliverable**: Secure platform with encryption, secret management, and authentication.
* **Resources**: DevOps Engineers, Security Analysts.

**6.2 Testing**

* **Objective**: Test the platform thoroughly for functionality, performance, and security.
* **Task**:
  + Conduct **unit testing** for individual components (AI models, dashboards, case management).
  + Perform **integration testing** to ensure that modules work seamlessly together.
  + Conduct **penetration testing** to identify and fix security vulnerabilities.
* **Deliverable**: Fully tested platform ready for deployment.
* **Resources**: QA Engineers, Security Analysts, Developers.

**6.3 Deployment**

* **Objective**: Deploy the platform to production and set up monitoring tools.
* **Task**:
  + Containerize the application using **Docker** and orchestrate deployments with **Kubernetes** for scalability.
  + Set up **NGINX** as a reverse proxy for load balancing.
  + Implement **real-time monitoring** of system performance using tools like **Prometheus** and **Grafana**.
* **Deliverable**: Production-ready deployment with real-time monitoring in place.
* **Resources**: DevOps Engineers.

**Technology Stack**

**Backend:**

* **Python (Django/Flask)**: Core API services, backend logic.
* **Node.js**: Real-time features and transaction streaming.
* **PostgreSQL**: Database for storing user and transaction data.
* **Redis**: In-memory data store for real-time alerting and notifications.
* **TensorFlow/PyTorch**: AI/ML models for risk assessment, fraud detection.

**Frontend:**

* **React.js/Next.js**: For interactive dashboards and frontend development.
* **D3.js**: For dynamic data visualizations.
* **ElasticSearch**: For semantic search functionality.
* **Tailwind CSS**: For building responsive and modern UI components.

**AI & Data:**

* **TensorFlow/PyTorch**: For developing machine learning models that analyze transaction patterns, customer behaviors, and risk levels.
* **Scikit-learn**: To build AI models for identifying suspicious activity and typologies like layering, smurfing, and rapid crypto transfers.
* **NLP (Natural Language Processing) Models**: For implementing semantic search and AI-driven report generation.
* **Chainalysis/Crystal Blockchain API**: For monitoring and risk assessment of cryptocurrency transactions and blockchain analysis.
* **Fireblocks API**: For handling secure crypto transaction processes, including minting, burning, and secure payments.
* **ElasticSearch**: For full-text search and semantic query functionalities.

**DevOps & Security:**

* **Docker**: For containerization, making the system portable and scalable.
* **Kubernetes**: For managing and scaling the deployment of containerized applications.
* **NGINX**: For load balancing and reverse proxy setup.
* **Prometheus/Grafana**: For real-time system performance and monitoring dashboards.
* **Vault by HashiCorp**: For managing sensitive keys, API tokens, and encrypted data.
* **OpenSSL**: For data encryption to secure sensitive transaction and user data.
* **OpenID Connect/OAuth2.0**: For user authentication and secure access control.

**Resource Allocation**

**1. Team Structure**

**Core Development Team:**

* **Backend Developers** (3): Handle API development, integration with AI models, transaction processing, and real-time alerting.
* **Frontend Developers** (2): Build the user interface, including dashboards, reporting modules, and case management screens.
* **AI/ML Engineers** (2): Develop and integrate AI models for fraud detection, AML typologies, and transaction monitoring.
* **DevOps Engineers** (2): Manage deployment, scaling, and security for the application using Docker, Kubernetes, and NGINX.
* **QA Engineers** (2): Responsible for testing the system functionality, security, and performance across all modules.

**Specialized Teams:**

* **Data Scientists** (1): Work on data analytics, training AI models, and developing risk profiles for suspicious activities.
* **UX/UI Designers** (1): Design user-friendly interfaces, wireframes, and visual elements for the compliance officers and admins.
* **Blockchain Developers** (1): Build integration with blockchain APIs and cryptocurrency transaction monitoring systems.
* **Security Analysts** (1): Ensure the security of data and transactions by performing vulnerability assessments and implementing encryption.

**Detailed Timeline**

**Phase 1: Planning and Architecture (Weeks 1-2)**

* Feature prioritization and architecture design.
* **Deliverables**: Product roadmap, architecture diagrams.

**Phase 2: UX/UI and Wireframes (Weeks 2-4)**

* User journeys, interface wireframes, and visual design.
* **Deliverables**: Wireframes and UI/UX prototype.

**Phase 3: Backend Development (Weeks 4-12)**

* API setup, database integration, AI model development.
* **Deliverables**: Backend APIs, AI models for transaction monitoring.

**Phase 4: Frontend and Dashboard Development (Weeks 6-14)**

* Dashboard implementation, reporting module, and admin panels.
* **Deliverables**: Interactive dashboards, reporting functionalities.

**Phase 5: Case Management and Alerts (Weeks 10-16)**

* Case management workflows, real-time alert system, escalation mechanism.
* **Deliverables**: Case management system, AI-powered alert generation.

**Phase 6: Security, Testing, and Deployment (Weeks 16-20)**

* Security setup, testing for performance and security, deployment to production.
* **Deliverables**: Secure platform, full testing results, production deployment.

**Step-by-Step Instructions for Product Development**

**1. Planning & Documentation**

* **Step 1**: Finalize product specifications and align with business objectives from CTG.
* **Step 2**: Prepare the technical architecture and confirm the technology stack based on scalability and security needs.
* **Step 3**: Define timelines and milestones with the product owner.

**2. Backend Development**

* **Step 4**: Set up the environment for API development using Python (Django/Flask) and begin work on core transaction processing APIs.
* **Step 5**: Integrate with cryptocurrency transaction monitoring APIs (Chainalysis, Fireblocks).
* **Step 6**: Develop machine learning models for AML/CTF monitoring and integrate them with the backend APIs.

**3. Frontend & Dashboard Development**

* **Step 7**: Use React.js for the frontend interface and D3.js for real-time data visualizations on dashboards.
* **Step 8**: Build the reporting module to generate compliance reports such as IFTI, SMR, and TTR.
* **Step 9**: Implement the semantic search functionality using ElasticSearch and integrate NLP for query handling.

**4. Case Management & Alerts**

* **Step 10**: Develop case creation and escalation workflows, including the ability to assign cases and track actions.
* **Step 11**: Implement the AI-powered alerts system to detect suspicious transactions in real-time and trigger escalation workflows.

**5. Security and Testing**

* **Step 12**: Implement encryption (OpenSSL) and secret management (HashiCorp Vault) to secure sensitive data.
* **Step 13**: Conduct rigorous unit and integration tests to ensure the system functions correctly.
* **Step 14**: Perform security audits and penetration testing before deployment.

**6. Deployment**

* **Step 15**: Containerize the application using Docker, orchestrate with Kubernetes for scalability, and deploy on production servers.
* **Step 16**: Monitor performance using Prometheus and Grafana, and optimize the system based on feedback.

**Risk Management and Contingency Plan**

* **Risk**: Delays in AI model development due to data complexity.
  + **Mitigation**: Break AI development into smaller, manageable tasks and focus on model validation early in the process.
* **Risk**: Performance issues with real-time monitoring under high transaction volumes.
  + **Mitigation**: Implement Redis for caching and use load balancing techniques with NGINX and Kubernetes for scaling.
* **Risk**: Security vulnerabilities during data handling.
  + **Mitigation**: Enforce encryption standards and perform regular security audits during the development lifecycle.