ChatGPT

KPN Upstream Supply Chain Monitoring System – Development Plan

This document outlines an internal development plan for a system to monitor and evaluate KPN Plantations' upstream supply chain for compliance with the EU Deforestation Regulation (EUDR). Under EUDR, palm oil products must be **deforestation-free (no forest clearing after 31 Dec 2020), fully traceable to the plot of land, and produced in compliance with local laws**, with each shipment backed by a due diligence statement (DDS) [1]. To meet these obligations, the planned system will provide: (1) plot mapping with geospatial deforestation monitoring, (2) supply chain traceability from farm to mill to export (avoiding any unsourced mixing), (3) legality document verification for each supplier and plot, and (4) automatic DDS report generation for EU TRACES submission. The following sections detail the development methodology, system architecture, product requirements, and a 16-week implementation workplan.

Technical Methodology

Our development approach will be **agile and iterative**, ensuring fast feedback and adaptability throughout the project:

- **Agile Sprint Development:** We will use **2-week sprints** to incrementally build and refine features. At the start, a **product backlog** of user stories (e.g. "As a GIS officer, I can upload a farm polygon…") will be created and prioritized. Each sprint will include planning, development, a demo/review with stakeholders, and retrospective. This approach allows quick adjustments to KPN's feedback and evolving compliance needs.

- **Cross-Functional Team:** Assemble a team covering **backend, frontend, and GIS expertise**, plus a product owner (from KPN sustainability/compliance) and a Scrum Master/Tech Lead. The GIS specialist will ensure geospatial data accuracy; backend developers will handle database, server logic, and integration; frontend developers will create the user interface (web portal) including map visualizations; and a QA engineer will oversee testing. Close collaboration (daily stand-ups, code reviews) will ensure all components integrate smoothly.

- **Testing & Quality Assurance:** We adopt a **test-driven mindset**. Unit tests will be written for critical backend functions (e.g. polygon validation logic, report calculations) and frontend components. Integration testing will verify that, for example, uploading a polygon triggers proper storage and can be retrieved for a DDS report. **Automated QA** pipelines (CI/CD) will run test suites on each code merge. Each sprint's end will include a **user acceptance test** with sample data to confirm the features meet KPN's requirements and data integrity standards. We will also prepare test scenarios for regulatory edge cases (e.g. multiple farm sources in one export) to ensure compliance output is correct.

- **Data Integrity Principles:** Ensuring data accuracy and trustworthiness is paramount. The system will enforce validation rules and verification steps at data entry to maintain high integrity. For

example, farm polygon uploads will be checked for proper coordinate reference (WGS84 lat/long) and geometry errors; delivery records must reference a known farm ID, etc. We will implement an **audit trail** for key data: any changes to polygons, supplier info, or documents will be timestamped and attributed to a user, preserving historical versions. This creates an **immutable record** of supply chain data changes, which is crucial for compliance audits. In line with security best practices, all data will be transmitted and stored securely (encryption in transit and at rest). Finally, to prevent data loss, regular backups of databases and file storage will be scheduled, and changes will be done via transactions to maintain consistency.

- **Iterative Refinement and Feedback:** After an initial **MVP is delivered (around mid-project)** covering core functionalities (mapping, traceability linking, basic DDS output), we will seek intensive feedback from the internal sustainability team. Any usability issues or new compliance clarifications can then be addressed in subsequent sprints. This iterative refinement ensures the final product is user-friendly and meets the actual on-ground needs of KPN staff.

By following this agile, test-focused methodology with an emphasis on data integrity, the development will remain flexible yet controlled, leading to a reliable system ready for real-world compliance needs.

System Architecture and Design

We propose a **modular, scalable web-based architecture** that fits KPN's current IT setup and can evolve with future infrastructure changes. The system will be designed for deployment on **Indonesian servers or cloud region**, to comply with data sovereignty requirements (while Indonesian law allows private companies to host data abroad if authorities have access, KPN can ensure data stays within Indonesia for assurance [2]). The architecture emphasizes security (role-based access control), performance for geospatial data, and integration readiness. Key components are:

- **Client Application (Web Frontend):** A responsive **web application (HTML5/JavaScript)** will serve as the main user interface. This frontend will provide interactive maps and forms for data input. We will use a mapping library (such as **Leaflet or Mapbox GL**) to visualize farm polygons on satellite or street map layers. The UI will support **multilingual toggling (Bahasa Indonesia and English)** – all static text, labels, and messages will be internationalized so users can switch language as needed. User access will be controlled via login; each user's role (e.g. admin, mill manager, compliance officer) will determine what they can see or do. For instance, a mill manager might only see suppliers for their mill, whereas an admin sees all data. The frontend will communicate with the backend via secure APIs (HTTPS/JSON).

- **Backend Application (Server & API):** The backend will be built with a modern framework (e.g. **Node.js/Express or Python/FastAPI** for rapid development) exposing RESTful APIs. It will handle business logic for each module: uploading and validating geospatial data, linking supply chain entities, generating reports, and authenticating users. Where appropriate, microservices or modular service design will be used – for example, a dedicated service for **GIS processing** (coordinate conversions, deforestation analysis) and another for **report generation**. The backend will include integration connectors for external systems: e.g., it could call a satellite monitoring API to fetch deforestation alerts for a given polygon, or in the future connect to KPN's weighbridge/ERP system to ingest delivery data. The server will enforce **data validation rules** uniformly (duplicating some

front-end checks to prevent any bad data via API). We will also implement **role-based access control** on the server (beyond just the UI) to ensure data endpoints respect user permissions.

- **Database Layer:** A robust database will store all supply chain data. We propose using a **relational database with spatial extensions** (e.g. PostgreSQL with PostGIS) to handle the farm polygons and their attributes. This allows efficient geo-queries (e.g. point-in-polygon, area calculations) and is compatible with open standards. The schema will cover tables for Suppliers/Farm Plots, Deliveries/Batches, Mills, Shipments/Exports, Documents, and Users/Roles. Relationships will link a delivery record to the specific farm (or multiple farms) that supplied it, and link those deliveries to a mill and export shipment. This **entity linking** approach ensures that for any export lot, we can retrieve *all originating plot coordinates*, fulfilling the EUDR traceability requirement that all source plots be identified [3] [4]. Document files (land titles, permits, etc.) will be stored in a secure file storage (with references in the DB for each supplier/plot), possibly on a file server or object storage (if cloud, e.g. S3-compatible storage in an Indonesia region). The database will maintain **audit logs** as mentioned (either via history tables or a built-in auditing extension) to record changes. We will design the DB for **scalability** – it should handle increasing numbers of suppliers and deliveries as KPN expands, and potentially integrate new data (e.g. new commodities or new regions) without needing a complete redesign.

- **Geospatial & Deforestation Monitoring:** A distinctive aspect of the architecture is integration with geospatial analysis tools. The system will link each farm polygon with satellite monitoring data to detect deforestation. This could be implemented by integrating with existing forest alert systems (such as Global Forest Watch GLAD alerts or a provider like **nadar.earth** for automated monitoring). For example, a scheduled job could run monthly to cross-check each plot's coordinates against recent forest loss data; any alert (forest clearing after 2020 within the polygon) is stored or flagged in the database. This addresses the **"deforestation-free" verification** by leveraging satellite imagery [5]. The architecture might include a **GIS processing module** that can handle on-demand computations (e.g. if a user wants to see a time-lapse of land cover for a plot, or calculate how much of a plot was forested as of 2020). Initially, simple integration (e.g. calling an API for alerts) will be done, with the option to expand into more complex analysis (possibly using GIS libraries or machine learning for land cover change detection) as needed. All geospatial data will use standard coordinate systems (WGS84) for consistency. We will ensure any coordinate data input (even if in local projection) is converted and stored in the standard lat/long to meet EU reporting specs [6].

- **DDS Report Generator:** The system will include a component to compile **Due Diligence Statements (Annex II reports)** automatically. This generator will pull data from the DB (supplier info, plot coordinates, volumes, etc.) and format it according to the regulation's requirements. The EUDR Annex II requires details such as operator information, product description & HS code, quantity, country of origin, and **"geolocation of all plots"** of production [4]. Our generator will assemble all required fields for each export shipment. The output might be a PDF report resembling the Annex II template, and/or a structured data file ready to upload to the EU TRACES portal. (According to guidance, DDS must be submitted through the TRACES IT system [7] – we will ensure our output is **TRACES-compliant**, possibly using digital forms or XML if available). This component will be designed such that if the EU updates the format or if KPN needs to integrate directly via API, the changes are isolated to this module. Digital signatures or stamping can be integrated if needed for authenticity.

- **Infrastructure & Deployment:** The system will be deployable on **KPN's existing infrastructure** – likely a central server at HQ or a private cloud. We will containerize the application (using Docker) to make deployment flexible across environments (on-premises or cloud). For scalability and reliability, a multi-tier deployment is envisioned: e.g. a web server, an application server, and a database server separated. If using cloud, we recommend using a reputable cloud provider's Indonesia region (for example, AWS Jakarta region or Azure SEA) to comply with any data residency preferences. The architecture also considers **scalability**: as data or user counts grow, we can scale vertically (bigger server) or horizontally (e.g. separate microservices, add load balancer for multiple app instances). Logging and monitoring services will be in place (to track usage, errors, performance). Security measures include network firewalls, encryption keys management, and compliance with Indonesian **Personal Data Protection Law** for any personal data in the system. User-based access control is central – the design ensures that users only see and act on data permitted by their role, and all API calls will authenticate the user's token/session.

- **Integration and Future-Proofing:** The design remains **extensible**. If KPN later introduces new systems (e.g. an ERP module for plantations or a blockchain traceability system), our API-centric approach will allow easy integration. Connectors or data import/export utilities will be included so that, for instance, existing supplier master data can be imported from Excel/CSV initially, or daily FFB delivery records could be batch-imported from mill systems. Likewise, the system could push DDS data directly to TRACES via API in the future if that becomes available. We also keep in mind other regulations – the architecture could be adapted for other sustainability compliance (e.g. other commodity due diligence laws) by configuring new data fields or modules, without rebuilding from scratch [8].

Overall, this architecture is tailored for KPN's needs: it leverages current technologies in a way that **fits the on-ground realities of plantation data**, and it is robust against regulatory changes. The design allows KPN to confidently trace every ton of CPO back to its exact farm, monitor for deforestation via satellite, ensure all legality documents are in order, and produce the required compliance reports – all within a secure, user-friendly system.

Product Requirements Document (PRD)

Below is the detailed Product Requirements Document, listing each major feature of the system along with its purpose, functionality, development approach, estimated implementation duration, and example user flow. This PRD serves as a blueprint for the development team to understand **what** needs to be built and **why**, and how users will interact with each component.

- **Plot Mapping Module**:
- **Purpose:** Establish a registry of all farm plots supplying KPN, with precise geospatial data, to enable deforestation risk monitoring and traceability down to the farm level. This addresses EUDR's requirement for **plot-level geolocation** for all raw material sources [9] [5].
- **Functionality:** Users (e.g. GIS officers or sustainability staff) can upload farm boundary polygons (in formats like Shapefile, KML, or GeoJSON) or draw them on an interactive map. The system standardizes the geometry to a consistent coordinate system (WGS84 lat/long) and runs QA checks: e.g. ensure no self-intersecting boundaries, polygon is within expected regions, and area is within plausible range for a farm. Metadata such as Farm/Supplier ID, name, and area (auto-calculated) are stored. The module links each polygon to its owner/supplier record. It also interfaces with satellite

data – for instance, upon upload the system could display recent satellite imagery of that plot and indicate if any forest clearing is detected within it post-2020 (through an automated query to deforestation alert data). Users can visualize all plots on a map layer, toggle background layers (satellite, terrain, etc.), and see at a glance which plots have deforestation alerts or overlaps with protected areas (flagged in the UI).

- **Development Method:** Use **PostGIS** to store polygon geometries and perform spatial validations (e.g. `ST_IsValid`, `ST_Area` for size, `ST_Intersects` to check overlap with other plots or forbidden zones). The frontend will use a map library (Leaflet with a draw plugin, or OpenLayers) to allow drawing/uploading shapes and previewing them on a map. We will implement server-side validation in the backend API to handle uploaded files: converting to GeoJSON, re-projecting to WGS84 if needed, and running QA rules (coordinate format, no crazy out-of-bound coordinates, etc.). For linking to satellite monitoring, we might integrate an API like **Global Forest Watch** or a local GIS tool: e.g. a background job could fetch forest cover loss data for the polygon's bounding box. Initially, a simple approach is to mark each plot with a status: "No deforestation detected" vs "Alert: possible clearing in 2022" based on available datasets. This can be refined over time.
- **Duration:** ~3 weeks for initial version. Week 1: set up spatial DB and basic upload API; Week 2: integrate frontend map UI and file upload; Week 3: implement advanced validations and link a sample satellite alert check, plus testing. Future sprints can enhance with more analytics, but a basic working module is achievable in this timeframe.

- **User Flow:** *Example:* A sustainability officer logs in and navigates to "Plot Management." They click "Add New Plot," fill in the farm name and owner info, and upload a KML file exported from a GPS survey. The system processes the file and displays the polygon outline on the map. The user sees that the polygon's area is, say, 50 hectares (auto-calculated) and that it's flagged as **"Low Risk: no deforestation alerts"** (the system checked that no forest loss has been recorded in this polygon since 2021). If the coordinate file was in a wrong format or the polygon self-intersected, the system would show a validation error for correction. The officer saves the plot, which is now stored and visible on the main map. Later, they can click the plot on the map to view its details: supplier name, area, coordinates, and a timeline of satellite-based forest cover (to visually verify no deforestation). All of this info will be used downstream in linking to deliveries and in DDS reports.

- **Supply Chain Traceability Module**:

- **Purpose:** Track the journey of palm oil from each farm plot through mills to export shipments, enabling **full traceability** as required by EUDR (no mass-balance obscurity). Each batch of product must be linked to its origin plots to prove it's deforestation-free [3]. This module ensures that **no unknown or non-compliant sources are mixed** into export lots, or if multiple sources exist, they're all accounted for and approved [10] [3].
- **Functionality:** This is the core data linking feature. Users (e.g. mill staff or supply chain managers) will record each delivery of Fresh Fruit Bunches (FFB) or CPO with its source. For instance, when a truck of FFB arrives at a mill, the user selects the supplying farm (from the registered plots in the system) and records the weight/quantity. The system creates a **delivery record** linking that farm to the mill and batch. Multiple deliveries in a day get processed into a **CPO lot** at the mill. The system can aggregate deliveries by day or batch number to form a production lot, which is assigned a lot ID. When an export shipment is created (e.g. a tanker or container of CPO), the user links the shipment to one or more lot IDs (and hence to the underlying farm sources). The module must prevent improper mixing – e.g., it will warn if someone tries to add a farm that is not approved (say a farm without legal documents or with a deforestation alert) into a lot destined for EU export. All

traceability data is stored, allowing queries like “show all farms that contributed to shipment X” or “list all shipments (and volumes) coming from farm Y”. We will avoid the simplification of mass-balance by **maintaining individual source identities**: even if a tank contains oil from 100 smallholders, the system will list all 100 source plots for that batch [10].

- **Development Method:** We will design the database relationships to model this many-to-many linkage (one shipment contains oil from many farms; one farm can supply many shipments). Likely we will have tables such as `Delivery` (with fields: date, farm_id, mill_id, weight, etc.), `Lot` (mill processing lot, with possibly blending of deliveries on a given day or tank), and `Shipment` (export event, linking to one or more lots). The backend will provide endpoints to create these records and enforce rules. For example, when closing a shipment record, the system can automatically compile the list of distinct farm polygons involved. We might implement some business logic like: only allow linking a farm to a mill if that farm is registered to supply that mill (to avoid data entry mistakes), or ensure volume consistency (sum of deliveries equals lot volume, etc.). The frontend UI will likely have forms and maybe drag-and-drop or selection interfaces. For instance, a mill manager might go to a “Deliveries” screen, input daily deliveries, then go to “Shipments” and create a new shipment that includes certain lots via multi-select. Given traceability is critical, we’ll also incorporate **visualization**: for a given shipment, the UI could show a map highlighting all source farm locations, to quickly see their distribution and risk profile. This helps the compliance team check if any high-risk area is included.
- **Duration:** ~4 weeks (perhaps split into two sprints). First, implement basic data model and CRUD for deliveries and shipments (2 weeks). Next, add business rules, aggregation logic, and front-end linking interfaces (2 more weeks). Additional time in later sprints can refine the UX or integrate automatic data import from existing systems (if KPN has an Excel of daily mill receipts, for example).

- **User Flow:** *Example:* A mill clerk opens the “Traceability” interface. They select their mill and date, then enter three FFB deliveries: 10 tons from Farm A (plot code #A123), 8 tons from Farm B (#B200), 5 tons from Farm A again later that day. The system logs each delivery with a timestamp and links to those farm IDs. That evening, the mill produces a batch of CPO (Lot L-001) from the day’s 23 tons. The clerk marks “Close Lot L-001” in the system, which then knows L-001 comprises Farm A and B’s produce (it can list A and B with respective tonnages). When it’s time to ship, the logistics staff creates a new Export Shipment record “March Export #1” and assigns it to Lot L-001 (and any other lots being shipped). The system now associates that shipment with Farm A and Farm B’s polygons. The compliance manager can later click on “March Export #1” and see a **traceability report**: it shows the lot’s total volume, the mill, and each contributing farm (with coordinates). Because every farm is known and mapped, if, say, Farm B had an unresolved deforestation alert, the system would have flagged it and the manager could remove that batch from the EU-bound shipment (ensuring non-compliant product is segregated as required [11]). In this way, the module guides users to maintain strict segregation of compliant material.

- **Legality & Compliance Document Management**:

- **Purpose:** Ensure each supplier (farm/plantation) and plot has all required legal and sustainability documents on file, verified and up-to-date. EUDR not only mandates deforestation-free sourcing but also that products are produced in accordance with the laws of the country of origin [12]. This means KPN must exercise due diligence that each farm has legitimate land rights, permits, environmental compliance, and has respected community rights (FPIC) etc. This module provides a structured repository and checklist for those documents, forming the evidence base for “legally produced” claims.

- **Functionality:** For each **supplier or plot**, users can upload and manage documents such as: land title or deed, plantation permit (e.g. HGU in Indonesia), environmental impact assessment (AMDAL) approvals, Free Prior Informed Consent (FPIC) documentation if applicable, business licenses, and any relevant certifications (e.g. ISPO/RSPO certificates). The module will categorize documents and track their status (e.g. *Pending Verification*, *Verified*, *Expired*). There will be a defined list of required docs per supplier/plot – possibly with variation for smallholders vs companies. The system can have a **checklist UI**: e.g., Supplier X – 8 of 10 required documents uploaded, 2 missing (highlighted in red). Admin users (compliance officers) can review uploaded files and mark them as "verified" (meaning they have been checked and are valid). If a document has an expiry (say a permit expiring in 2025), the system should flag when it's nearing expiry. To maintain integrity, once a document is verified, any subsequent re-upload could require re-verification (to avoid someone swapping a document unnoticed). This repository should also allow quick retrieval: e.g., the compliance team can filter suppliers to find any that lack a particular certificate or quickly pull up the legal docs for an audit. The module might also integrate with **OCR** or text search to extract key info (optional, if time permits) – for example, automatically reading a land title to record the owner name or area. However, initially it can be meta-data driven (user enters key info manually alongside the upload).
- **Development Method:** We will create a Document entity linked to Supplier/Plot records in the database. The files will be stored in a secure storage (file system or cloud bucket). We'll implement file upload APIs with virus scan and size/type validations (only PDF, image files allowed, no executables). A set of required document types can be configured (perhaps a lookup table or JSON config). The frontend will have an interface under each supplier profile to **upload files** to specific categories (e.g. a form with fields: select document type from dropdown, attach file, enter reference number or date). A preview or download link will be available for each saved file. We will include a simple workflow: a newly uploaded doc is tagged "Pending"; an authorized user can click "Verify" after reviewing it, which stamps their user ID and date on it as verified. We may also allow marking "Rejected" if the document is illegible or wrong, prompting the supplier team to upload a correct one. Notifications can be part of this – e.g. if a required doc is missing or expiring, the system can list these as tasks for follow-up. Development will utilize existing libraries for file handling and possibly an OCR service if needed for text (though that may be a stretch goal). The focus is on secure storage and easy retrieval.
- **Duration:** ~3 weeks. The first 2 weeks to implement basic file upload/download and linking to supplier records, and a simple UI to view and manage documents. The third week to add verification workflow, status tracking, and testing the permission controls (ensuring only authorized roles can verify or view sensitive docs). Additional enhancements (OCR, automated reminders) could be scheduled in a later sprint if time allows or in future versions.

- **User Flow:** *Example:* A compliance officer opens the profile for Farm A (Supplier #A123). On the "Documents" tab, they see a list of document categories: *Land Title*, *Operating Permit*, *Environment License*, *FPIC*, etc., each showing a status icon (green check for verified, red X for missing, orange clock for expiring soon). For Farm A, suppose the Land Title and Environment License are uploaded and verified, but the FPIC document is missing. The officer contacts the field team, who then logs in and uploads a scanned PDF of the FPIC consent form for Farm A. The system stores this and marks FPIC as "Pending Verification". The compliance officer gets a notification or sees the pending status, opens the file, checks it for authenticity, and then clicks "Verify", adding a note "Verified by [Name] on Aug 10, 2025". Now FPIC shows as verified (green). Later, when preparing a DDS or during an external audit, the officer can quickly retrieve all of Farm A's documents from the system to demonstrate that Farm A had all legal requirements fulfilled at time of sourcing. If a required

document was not on file, the system would highlight that Farm A is “not compliant” in the supplier list, prompting action before any export including that farm’s product. This module ensures **no supplier falls through the cracks in legal compliance** checking.

- **Due Diligence Statement (DDS) Generation**:

- **Purpose:** Automate the creation of the **Annex II DDS reports** required by EUDR for each shipment, ensuring that all necessary information is compiled accurately from the system’s data. This saves time and reduces error in preparing compliance documentation for EU customs. The DDS needs to affirm the shipment is deforestation-free and legal, including details on volumes, origin plots, and supplier info [4] [9]. By auto-generating this, we support swift submission to the EU’s TRACES platform and maintain consistency across all exports.
- **Functionality:** When an export shipment is ready to be sent to the EU, the user can trigger a “Generate DDS” action. The system will gather all relevant data for that shipment: operator details (KPN’s info, registration/EORI number, etc., likely entered in a settings page), the commodity and product details (e.g. “Crude Palm Oil, HS code 151110”), quantity (net weight of the shipment), and the **list of geolocations (latitude/longitude or polygon identifiers) for each farm involved**. It will also include the country of production (Indonesia) and confirmation that production did not occur on deforested land after 2020 (this can be a statement automatically included, assuming all linked farms have no deforestation flags). If any risk assessment info or mitigation steps are needed for completeness, those can be incorporated (for instance, a section stating “Risk assessment conducted – all source plots in low-risk areas or risk mitigated”). The output format will follow the Annex II structure – likely a PDF form or an electronic form. The system might provide a preview for the user to review. Once finalized, the DDS can be **downloaded** and then uploaded to the EU TRACES portal, or potentially directly transmitted if integration is achievable. Additionally, the system will store a copy of each DDS generated for record-keeping, along with a unique reference. Because TRACES, upon submission, returns a **Reference Number and Verification Code** as proof of submission [13] [14], we may include a field for the user to enter those once they have submitted the DDS. That way the system maintains a link between the shipment and the official DDS reference number, which could be useful for future auditing or chain communication.
- **Development Method:** We will create a service or function in the backend that collates data from various tables (as outlined above) and fills a **report template**. For the template, we might use a PDF generation library (like ReportLab for Python or PDFMake for Node) or simply generate a structured HTML/PDF. We’ll ensure the format meets the known Annex II specifications (as per the regulation or guidance documents). For example, Annex II includes sections for operator info, product info, geolocation info, etc., so our code will populate each section. We will validate that **all required fields are present**; if some data is missing (e.g. a supplier missing coordinates or legal docs), the generation function will throw an error or warning so the user knows to fix data before submission. The development will involve building this data aggregation logic and formatting it correctly. We’ll also implement a feature to log the generated statement in the database (with timestamp and possibly a copy of the file or a pointer to it). If time permits, we could explore using the TRACES system’s API (if available) to submit directly – however, since it might not be open, the primary approach is to generate a file for manual submission. Testing this feature will involve comparing the output against the regulation checklist to ensure compliance.
- **Duration:** ~2 weeks for initial development (likely one sprint towards the latter half of the project). This includes building and testing the template with one commodity (palm oil). As KPN deals primarily with palm oil, one template suffices; if in future they needed to include other commodities,

the template system could be extended. One week to implement the data gathering and PDF creation, and another week to refine formatting, add any minor fields and perform test runs with sample data.

- **User Flow:** *Example:* A logistics coordinator has finalized an export "March Export #1" which is scheduled to ship to Rotterdam. They open that shipment record in the system and click "Generate DDS". A modal or page appears showing the draft DDS: it automatically fills in KPN Plantation's name, address and EORI number (saved in system settings), selects "Export" as activity type, lists the product as *Palm Oil – HS 151110* with quantity *30,000 kg*, and lists **5 geolocation points** (since suppose the shipment includes oil from 5 distinct farm plots). Each geolocation might be represented by coordinates or an identifier that maps to a coordinate file attached. The coordinator reviews the info. They notice all looks good – the system even included a statement "All 5 source plots (listed above) are verified to be deforestation-free (no forest loss after 2020) and compliant with Indonesian law [1] [15]." The coordinator clicks "Download DDS" and gets a PDF. They then log into the EU TRACES portal and upload this DDS file as part of the shipment declaration. After submission, TRACES returns a Reference Number and Verification Code (displayed to the user on that portal). The coordinator goes back into our system, enters these codes into the shipment record for completeness. The shipment status in our system might be marked "DDS Submitted". During an EU customs check, if needed, KPN can quickly retrieve this DDS from the system. This automation drastically reduces manual effort – previously, they might have compiled such a report by hand for each shipment, but now it's done with one click, pulling accurate data directly from the source records.

- **User Management & Security**:

- **Purpose:** Control access to the system's data and features via user accounts, roles, and permissions. Given the sensitive nature of compliance data (e.g. personal details of smallholders, internal supply chain info), we must ensure only authorized personnel can view or edit certain information. Also, multilingual support ties into user preferences or settings.
- **Functionality:** The system will have a user authentication mechanism (login with username/password, and possibly SSO integration if KPN has an existing directory). There will be defined **roles** such as *Administrator*, *Compliance Manager*, *GIS Officer*, *Mill Manager*, etc., each with specific permissions. For example, an Admin can manage users and all data; a Compliance Manager can view everything and approve documents but maybe not manage users; a Mill Manager can input deliveries for their mill but not see other mills or edit legal documents; a Read-Only Auditor role might be able to view data and generated reports but not change anything. We will implement role-based access control (RBAC) in the backend – each API endpoint will check the user's role and data scope. On the frontend, UI elements (menus, buttons) will be shown/hidden based on permissions. This feature includes a user management UI for admins to create or deactivate accounts, assign roles, and possibly reset passwords. Regarding **multilingual support**, the interface text will be available in at least English and Bahasa Indonesia. Users can select their preferred language (or it's set by admin per account). All labels, button texts, and notifications will come from a translation file for the chosen locale. The content like user-entered data (farm names, documents) remain as entered, but system messages will translate. This ensures both local staff and any expat or international staff can use the system comfortably.
- **Development Method:** We will likely use an off-the-shelf authentication system or framework module (for example, JWT tokens for session management on the API, and a library for password hashing and verification). User roles and permissions will be stored in the database (e.g. a `Users`

table and a `Roles` table with mappings). We'll seed the system with default roles and allow customization if needed (though fixed roles may suffice initially). We will integrate **access control checks** in each feature's backend logic. For instance, the "upload polygon" API will require a role with GIS edit permissions; if a user without that tries, the system returns an unauthorized error. Similarly, document verification actions might be limited to Compliance role. On the frontend, we can manage this via route guards and conditional rendering depending on the user's role stored in their session. For internationalization (i18n), we'll use a standard i18n library (such as i18next for JavaScript) and maintain JSON/YAML resource files for en and id languages. All UI text will reference keys from these files. Testing will involve creating test users with various roles to ensure the permission matrix is correctly enforced. We also plan to incorporate **audit logging of user activities** (who logged in, who made changes) as part of security monitoring.

- **Duration:** ~2 weeks (likely spread in parallel with other development). Basic authentication and user model can be set up in Week 1 of development (so that the system is secure from the start). Defining roles and implementing fine-grained permissions and the user admin screens might take another week. Multilingual text integration can be done gradually as the UI is built, but ensuring complete i18n coverage and proper testing may take a few days toward the end.
- **User Flow:** *Example:* An Admin user logs into the system via the login page. After the initial setup, they create accounts for other team members: e.g. they add a user "JohnDoe" with role "Mill Manager" and assign them to Mill #3. JohnDoe logs in and the system recognizes his role and mill assignment, so when he uses the system, he only sees data relevant to Mill #3 (deliveries and shipments for that mill). JohnDoe does not even see the menu items for "Admin Panel" or "User Management" or "Compliance Documents" because those are outside his role. Meanwhile, a Compliance Manager user logs in and has access to the "Documents" section to verify legal files and the "Reports" section to generate DDS. She prefers to use the system in Bahasa Indonesia, so she switches the language toggle in the top menu – instantly, the interface labels change to Indonesian (the data like farm names remain as is). She can comfortably navigate the system in the local language. All these interactions are secure: if JohnDoe tried to access an admin URL manually, the backend would reject it. And every action (like uploading a document or editing a delivery) is tracked with the user's ID, providing accountability. This feature ensures the system is **used only by authorized personnel in their respective scopes**, and it supports KPN's bilingual operational environment.

Each feature above is aligned with the overall goal of EUDR compliance and has been specified with clear purpose and development details. The PRD will guide the team during implementation, and any changes will be documented through agile iterations.

Implementation Workplan (12–16 Week Timeline)

The project will be executed over approximately 16 weeks, broken into 8 two-week sprints. Below is a week-by-week plan with sprint focuses, deliverables, responsible roles, and milestones. This schedule is designed to deliver a functional Minimum Viable Product (MVP) by around week 12, with additional time for testing, feedback incorporation, and final deployment by week 16.

- **Sprint 1 (Week 1–2) – Project Initiation & Design**
- *Focus:* Project kickoff, requirements alignment, and foundational setup. The team will review the PRD and refine user stories. Architecture and tech stack decisions are finalized. Development environments are prepared.

- *Expected Output:* **Technical architecture document** (finalized based on above design) and a product backlog of all features/tasks. Environment set-up completed (servers provisioned, repositories created, CI pipeline configured). A basic skeleton of the application (empty modules, base project structure) is created.
- *Roles:* Solution Architect & Product Manager lead the architecture finalization and backlog grooming. DevOps/Backend Engineer sets up servers and CI. All team members participate in kickoff meetings (understanding roles and domain context).

- *Milestones:* **Architecture & Design Sign-off** – by end of Week 2, stakeholders approve the system design and infrastructure plan. **Project Backlog Complete** – user stories for at least the first few sprints are well-defined and estimated.

- **Sprint 2 (Week 3–4) – Plot Mapping MVP**

- *Focus:* Implementing the Plot Mapping Module's core functionality. This includes the backend API for polygon upload/storage and the frontend map interface to display and add plots. Basic validation of polygons is included.
- *Expected Output:* **Geospatial Upload & View Feature:** Users can upload a farm polygon and see it on a map in the web interface. The polygon data is saved in the database. We will also integrate a base map (satellite imagery layer) and show polygon attributes (area calculated, etc.). Some QA rules (valid geometry, coordinate conversion) will be working, though possibly not all edge cases yet.
- *Roles:* GIS Developer (back-end) implements spatial DB model and validation logic; Frontend Developer builds the map UI and upload form; Backend Developer creates API endpoints and ties in PostGIS. QA begins writing test cases for file upload.

- *Milestones:* **Demo of Plot Upload** – by end of Week 4, we can demo adding a farm polygon and viewing it. This is a significant milestone as it visually shows the system working. Any initial feedback from GIS teams (e.g. on coordinate accuracy or UI ease) is collected for next iteration.

- **Sprint 3 (Week 5–6) – Traceability Data Model & Input**

- *Focus:* Building the Supply Chain Linkage module. This sprint establishes the database tables and basic interfaces for deliveries and shipments, linking to farms and mills. Possibly a simple UI to input delivery records is provided.
- *Expected Output:* **Traceability data structure:** The system can record a delivery of crop from a farm to a mill and create a shipment entry. We may not have a pretty UI for all of it yet, but the backend can store and retrieve these records. A rudimentary frontend form or admin interface allows adding deliveries and creating a shipment (for testing the flow). This might be initially done in a simplified manner (assuming one delivery per lot, etc.) and then expanded.
- *Roles:* Backend Developer designs and creates the **Delivery**, **Lot**, **Shipment** tables and relationships. They implement core logic for linking farm -> delivery -> shipment. Frontend Developer sets up basic pages or forms for these operations (or uses a temporary admin UI if needed). The Product Manager/BA ensures the logic aligns with real workflows at KPN. GIS support may be minimal here, but they ensure that linking works with the polygons.

- *Milestones:* **End-to-end Data Flow (prototype)** – by Week 6, we can enter a delivery and mark it as shipped, and then query which farm that shipment came from. Internally, this proves the traceability

chain data model works. This will likely be an MVP demonstration (e.g. via API or a simple form output) that sets the stage for the more user-friendly interface in the next sprint.

- **Sprint 4 (Week 7–8) – User Interface & Business Rules for Traceability**

- *Focus:* Enhance the Supply Chain module with a user-friendly UI and enforce business rules (no mixing of unapproved sources, volume summaries, etc.). Also, start integrating the plot mapping with traceability (e.g., select from existing plots when recording a delivery).
- *Expected Output:* **Traceability UI and Validation:** A proper **"Traceability" screen** or set of screens is available. For example, a page for mill managers to input daily FFB deliveries (with dropdown of supplier plots), and a page for creating/viewing shipments. The system now has controls like preventing selection of a farm that isn't verified, or warning if a delivery is added with no corresponding plot. We also show aggregated info (e.g. listing all farm sources on a shipment view). The link between map and traceability might be established: clicking a plot could show its delivery history, etc., though that could be iterative. Basic constraints such as "shipment cannot be closed if any source farm is flagged non-compliant" might be implemented.
- *Roles:* Frontend Developer focuses on building intuitive forms and tables for deliveries/shipments. Backend Developer refines the logic (e.g. implement a rule that checks document status of a farm before allowing it in a shipment). The GIS Developer helps ensure that farm selection in forms is tied to spatial data correctly (maybe via farm ID). QA now can test a real scenario: add some farms, deliveries, shipments and verify the data integrity and UI flows.

- *Milestones:* **Usable Traceability Module** – end of Week 8, KPN team members can try using the system for a small test case: registering a couple of farms, simulating a day's deliveries and generating a dummy shipment, and see the output. Milestone is that internal users find the interface making sense and the data linking correctly. Adjustments from their feedback will be noted for coming sprints. Also by this point, **mid-project review** occurs (around halfway): ensuring we are on track and adjusting scope or priorities if needed.

- **Sprint 5 (Week 9–10) – Document Management & Verification Workflow**

- *Focus:* Develop the Legality Assessment module where documents are uploaded and verified for suppliers/plots. Also integrate document status checks into other parts (e.g., blocking unverified suppliers from being used in shipments if required).
- *Expected Output:* **Document Repository Feature:** Users can go to a supplier's profile and upload required documents. There will be a list of document types, and the UI to upload, view, and mark verification. By end of Week 10, the system can store files securely and display their statuses. In addition, some automation: for instance, when a farm's required documents are all marked verified, that farm is tagged as "Legally OK" in the database. This status can reflect in the traceability module (like a green check icon next to farm name in selection lists, vs a red icon if something's missing). We should also implement at least one notification or report: e.g. a view or export of all farms and whether they have complete documentation, so compliance officers can act on any gaps.
- *Roles:* Backend Developer sets up file storage and document metadata tables. Frontend Developer creates the document upload interface and listing. The security specialist ensures access control on documents (maybe only compliance role can download sensitive docs). QA tests file upload with various sizes/types and ensures unauthorized roles cannot access restricted functions. The Product

Manager ensures that the list of required documents is configured correctly per KPN's compliance checklist.

- *Milestones:* **Compliance Data Integration** – by Week 10, the **legality status** of suppliers is being captured. Milestone: we can show that if a farm is missing a document, the system knows it (and possibly reflects that in traceability, e.g., preventing usage or at least flagging). Also, a **Document Upload Demo** will be done to stakeholders, showing how a user uploads and an admin verifies a document. Feedback on ease-of-use (especially important because this might involve many files) will be gathered.

- **Sprint 6 (Week 11–12) – DDS Report Generation & Final Feature Integration**

- *Focus:* Build the Due Diligence Statement generator and ensure all pieces of the system work together. This sprint is about completing the core functional scope. It also includes finishing touches on multilingual support and any remaining user management tasks (so that by now, all intended features exist in basic form).
- *Expected Output:* **DDS Generation Tool:** By end of Week 12, a user can choose an export shipment and click "Generate DDS", and the system produces a populated report (e.g. downloadable PDF) containing required info (operator details, product, volume, farm geolocations, etc.). This output will be tested for compliance with Annex II format. Additionally, we aim for a **fully integrated system**: meaning a workflow starting from registering a farm, uploading its docs, recording deliveries, and producing a DDS for a shipment can be executed within the software. The multilingual UI should be mostly polished by now (all labels translated), and the user-role permissions enforced on all new features (e.g., only managers can generate DDS, etc.).
- *Roles:* Backend Developer works on the report compilation logic and PDF generation. They also cross-check the EU requirements to ensure nothing is missing. Frontend may implement a nice interface for the report generation (maybe a preview or a form to confirm details before generation). The Product Manager/Business Analyst will help verify that the DDS output meets the compliance team's expectations (perhaps comparing with a manual example). Meanwhile, any dangling tasks on user management and i18n from previous sprints are wrapped up by the respective developers. QA will thoroughly test a full scenario with dummy data across the modules, paying attention to data consistency (e.g., if a farm without coordinates somehow got into a shipment, does the DDS handle it or block it appropriately?).

- *Milestones:* **MVP Completion** – at the end of Week 12, we effectively have a Minimum Viable Product that covers all four major components and can fulfill the basic EUDR compliance workflow. A major milestone review will be held where the team presents generating a DDS from end-to-end data in the system. Achieving this by Week 12 meets our target and allows the remaining time for improvements and rollout prep.

- **Sprint 7 (Week 13–14) – Testing, Refinement & User Training**

- *Focus:* Intensive testing (both functional and user acceptance testing) and refinement of the system based on feedback. This includes bug fixing, performance tuning, and enhancing any UX issues discovered. Also, initial user training/documentation efforts start.
- *Expected Output:* By the end of this sprint, **all critical bugs are resolved** and the system should be stable. We expect to have a **beta release** deployed on a staging server for wider internal testing by KPN's team. We will also produce user guides or at least help documentation for the main features

(possibly simple how-to notes or tooltips in the UI). Any features that were minimal earlier can be refined now (for example, improving the map layer performance if many polygons, or making the DDS output format prettier). Performance testing with larger data sets might be done now to ensure the system can handle, say, hundreds of farms and daily deliveries. If any compliance requirement was missed or updated (for instance, if new guidelines from EU or issues from internal audit come up), we address them now. Additionally, we start conducting **training sessions** for key users (maybe workshop style demonstrations) so that by deployment, users are familiar with how to use the system.

- *Roles:* QA Lead coordinates full-system testing, creating test reports. Developers (backend & frontend) work on bug fixes and incremental improvements identified. The UI/UX designer (if present, or frontend dev) might refine styling and usability details. The Product Manager gathers feedback from test users (e.g. some KPN staff trying the system) and prioritizes any last-minute changes. A technical writer or team member will start drafting user manuals or inline help texts.

- *Milestones:* **User Acceptance Testing (UAT) Sign-off** – by Week 14, representatives of the end-users should have tested the system and confirmed it meets requirements for go-live. All high-priority bugs found in UAT are fixed or have work-arounds. Also, **Performance Test** results are reviewed to ensure the system doesn't break under expected load.

- **Sprint 8 (Week 15–16) – Deployment & Handover**

- *Focus:* Production deployment and project closure. This sprint covers migrating the final code to production environment, finalizing documentation, and a post-deployment support window. We also ensure compliance with Indonesian data regulations is documented and satisfied.
- *Expected Output:* **Production-ready System:** In week 15, we deploy the system on KPN's chosen production servers (ensuring it is in an Indonesia-based data center or cloud as planned). We populate it with initial master data (existing supplier list, etc., if provided by KPN) so it's ready to use. We also deliver the **Documentation** package: technical docs (for maintenance), user guide (for everyday users), and an administrator guide (covering user management, backups, etc.). Additionally, a **maintenance plan** is provided (how to handle updates, who to contact for support, etc.). We will conduct a final training session for all end-users just before launch. During Week 16 (and possibly a bit beyond as needed), the dev team will closely monitor the system in production – checking for any issues (error logs, performance metrics) and be on standby to fix any urgent post-launch bugs. After things stabilize, the project transitions to maintenance mode or to the in-house IT team.
- *Roles:* DevOps Engineer and Backend Developer handle the deployment process (setting up production database, environment variables, security config). The Solution Architect double-checks that all infrastructure (load balancers, firewalls, domain setup for the web app) is correctly configured. The compliance team and IT security may do a final audit (ensuring data is indeed stored as per sovereignty requirements, e.g. backups are also in-country, etc.). The Product Manager and technical writer finalize all documentation. All team members may help in training sessions or in handover discussions.
- *Milestones:* **Go-Live** – by mid-to-late Week 15, the system is live and accessible to the intended users at KPN. **Project Handover Complete** – by end of Week 16, KPN's internal team (or relevant staff) have been given the knowledge to operate and maintain the system, and all project deliverables are signed off. A retrospective is held to document lessons learned and ensure any pending nice-to-have features are noted for future improvements.

By following this workplan, we anticipate delivering a fully functional upstream supply chain monitoring and evaluation system within 4 months. The plan's built-in feedback loops, testing phases, and clear milestones will help keep the project on track and aligned with KPN Plantation's compliance objectives. With this system in place, KPN will be well-equipped to demonstrate EUDR compliance – from **precise plot mapping and continuous deforestation monitoring** [5], to **complete farm-to-port traceability** [16], thorough **legal due diligence records**, and instant generation of compliant **DDS reports** [16] for EU authorities. All data will be handled securely and in accordance with Indonesian regulations, and the bilingual, role-based design ensures it fits seamlessly into KPN's operational workflow.

Sources: The development plan has been informed by EUDR requirements and industry best practices. Notably, EUDR mandates traceability to plot-level coordinates and legality verification [1] [15], which this system directly addresses. Guidance from EU and industry (e.g. European Commission forums and compliance tool providers) reinforces the need for geolocation mapping, satellite monitoring, and risk mitigation in supply chains [5] [16]. Indonesian data protection guidelines were considered to ensure data sovereignty compliance [2]. These sources and KPN's internal policies collectively shaped the system's features and the implementation approach. The result is a tailored solution ready to support KPN Plantations in achieving deforestation-free, legally compliant palm oil supply for the EU market.

---

[1] [3] [5] [9] [12] [15] What is the EUDR: Understanding deforestation legislation
https://www.sap.com/latvia/resources/eu-deforestation-regulation-guide

[2] Data localization and regulation of non-personal data | Indonesia | Global Data and Cyber Handbook | Baker McKenzie Resource Hub
https://resourcehub.bakermckenzie.com/en/resources/global-data-and-cyber-handbook/asia-pacific/indonesia/topics/data-localization-and-regulation-of-non-personal-data

[4] [6] [7] [13] [14] Trusty | EUDR Compliance: Guide to the Due Diligence Statement
https://www.trusty.id/post/eudr-compliance-the-due-diligence-statement

[8] [16] Best 5 EUDR compliance tools for 2025 supply chain due diligence
https://www.coolset.com/academy/best-5-eudr-compliance-tools-for-2025-supply-chain-due-diligence

[10] [11] Traceability and geolocation of commodities subject to EUDR - European Commission
https://green-forum.ec.europa.eu/nature-and-biodiversity/deforestation-regulation-implementation/traceability-and-geolocation-commodities-subject-eudr_en