Arboracle: A Strategic Blueprint for a Minimum Lovable Product and Scalable Architecture

The Arboracle Vision: A Philosophy of Digital Ecology

This section establishes the foundational philosophy of Arboracle, moving beyond a simple feature list to create a powerful, mission-driven narrative. The proposed strategy defines a brand and product philosophy that is not just nature-themed, but deeply rooted in the principles of natural systems. This approach provides a unique and defensible market position that will resonate with users, communities, and investors.

Branding Through Biomimicry and Biophilia: Designing a Living Interface

To create a brand that is authentically aligned with nature and biodiversity, Arboracle's design language should be guided by the dual principles of biophilia and biomimicry. This methodology goes beyond superficial aesthetics, embedding the patterns and processes of the natural world into the core user experience. The result is an application that feels organic, intuitive, and alive, fostering a genuine connection between the user and the ecological data they are curating.

Biophilic Design Integration

Biophilic design is founded on the principle that humans have an innate connection to nature and that incorporating natural elements into our environments can reduce stress, improve focus, and enhance well-being.[1] For Arboracle, this translates into a user interface that is calming, engaging, and restorative.

- **Atmosphere and Imagery:** The application should utilize high-quality, immersive imagery of natural landscapes as backgrounds and key visual elements. Similar to how applications like Calm or Airbnb use visuals to evoke a sense of peace or adventure, Arboracle can use them to create a serene and focused environment for ecological observation.[1]

- **Natural Color Palettes:** The color scheme should be drawn directly from the natural world. Palettes such as Pantone's "Nature Remastered," with its mossy greens, earthy browns, and mineral tones, provide an excellent starting point.[2] These colors are inherently harmonious and easy on the eyes, reinforcing the app's connection to nature.[3] The interface could even adapt its palette based on the user's local ecosystem or the season.
- **Organic Forms and Textures:** The design should move away from rigid, geometric grids and embrace organic, flowing shapes and subtle textures that mimic natural materials like wood grain, stone, or leaf patterns.[1] This adds a tactile depth to the digital interface, making it feel less artificial and more grounded.[6]

Biomimicry as a Functional Principle

While biophilia addresses the look and feel, biomimicry informs the app's core functionality by emulating nature's time-tested processes and systems.[5] This strategic layer transforms Arboracle from a simple data tool into a piece of functioning "digital ecology."

- **Navigation and Information Flow:** User navigation can be inspired by natural patterns. For example, information could be structured in a branching pattern like a tree, or user flows could mimic the efficient, meandering path of a river, making the experience feel intuitive rather than mechanical.[9]
- **Adaptive Interfaces:** The application can be designed to adapt to user context and behavior, much like an organism responds to its environment.[5] The UI could subtly shift its appearance based on the time of day, local weather conditions, or the user's recent activities, creating a dynamic and responsive experience.
- **Data Organization and Visualization:** Complex datasets can be presented using structures found in nature. Fluid grid layouts that mimic natural tessellations (like a honeycomb) or fractal-based organizational systems can make dense information feel more comprehensible and less overwhelming.[9]

This approach—moving from a simple "nature theme" to a "nature-principled" design—is a significant strategic differentiator. It demonstrates a profound vision that is not easily replicated by competitors. This narrative of building a digital tool that functions in harmony with natural principles is highly compelling for users who care deeply about biodiversity and for investors seeking ventures with a unique, defensible market position. The branding for the Kunming-Montreal Global Biodiversity

Framework, for instance, uses Voronoi patterns to represent the interconnectedness of its 23 targets, visually communicating a complex system as a unified, natural whole.[10] Arboracle can adopt a similar ethos, ensuring its brand and product are one and the same.

The Minimum Lovable Product Proposition: The "Digital Ecologist" Journey

The Minimum Lovable Product (MLP) must focus on a single, compelling user journey that delivers immediate value and creates an emotional connection, turning early adopters into passionate advocates. The target user for the MLP is conceived as a "Digital Ecologist"—an individual actively engaged in mapping, understanding, and nurturing their immediate environment.

The MLP feature set is designed to support this journey:

1. **Hyper-Local Data Collection:** A streamlined, map-first interface for logging observations of trees and soil conditions. The design should prioritize simplicity and intuition for non-technical users, drawing inspiration from the field-ready tools of Open Foris Ground and GeoNature's mobile applications.[11]
2. **AI-Powered Identification & Analysis:** Immediate value is delivered through an integrated AI tool. Users can upload a photograph of a tree, and the application will provide a species identification and an initial health assessment, similar to the functionality of TreeTect.[13] To maintain scientific credibility, results will be presented with clear confidence scores, reflecting the precision and recall metrics of the underlying models.[14]
3. **The Living Map:** A personal, dynamic map that visualizes the user's collected data. This is the heart of the "lovable" experience. Instead of a static collection of pins, the map will be a living representation of the user's ecosystem, employing biophilic design principles. Visual cues on the map could change based on the health, age, and diversity of the cataloged life, providing a rich, at-a-glance summary.
4. **Ecosystem Services Insights:** To connect individual action to global impact, the app will provide simple, understandable calculations of the ecosystem benefits provided by the trees a user has mapped. This feature, inspired by OpenTreeMap, can quantify metrics like estimated carbon sequestration or stormwater runoff reduction, making the user's contribution tangible.[16]
5. **Community Seeding:** While a full social network is beyond the scope of the MLP,

a simple feature allowing users to share a unique link to their personal map or a specific observation is crucial. This plants the seed for future community-driven features, echoing the successful crowdsourcing model of platforms like OpenTreeMap.[17]

Architectural Foundations: Selecting the Open-Source Bedrock

The selection of a foundational open-source platform is a critical decision that will influence Arboracle's scalability, stability, and long-term viability. A rigorous analysis of the leading platforms in the biodiversity and urban forestry space reveals that a hybrid, service-oriented architecture is the optimal path forward, leveraging the strengths of established projects while maintaining the flexibility to innovate.

Comparative Analysis of Foundational Platforms: GeoNature, OpenTreeMap, & Open Foris

Three primary open-source ecosystems stand out as potential foundations: GeoNature, OpenTreeMap, and Open Foris. Each offers distinct advantages and disadvantages.

- **GeoNature:** A comprehensive biodiversity management system developed by French national parks.[12]
  - **Strengths:** It boasts a large, active community, ensuring long-term stability and support.[12] Its architecture is modern and modular, built on a Python/Flask backend, an Angular frontend, and a powerful PostgreSQL/PostGIS database.[18] It supports a wide range of biodiversity data (fauna, flora, habitats) and adheres to important data standards like FAIR (Findable, Accessible, Interoperable, Reusable).[12] A well-documented API is also available.[20]
  - **Weaknesses:** The frontend is built with Angular, which does not align with the recommendation for a Next.js/React stack. The community and documentation are predominantly French, which could present a language and support barrier.[21]
- **OpenTreeMap:** A platform specifically designed for crowdsourced urban tree

inventories.[17]

  - **Strengths:** Its mission is closely aligned with Arboracle's core focus on trees. It includes built-in ecosystem service calculations and has existing, albeit dated, mobile applications and an API.[16]
  - **Weaknesses:** The project appears to be largely dormant. An analysis of its GitHub repositories reveals minimal commit activity in recent years, indicating a lack of active development.[24] The technology stack is older (Python/Django), and its OpenSSF security scorecard is alarmingly low, making it a high-risk foundation for a new product.[24]
- **Open Foris:** A suite of powerful forestry tools backed by the Food and Agriculture Organization of the UN (FAO).[26]
  - **Strengths:** The FAO backing lends it immense credibility and resources. It is not a single platform but a collection of best-in-class tools for mobile data collection (Ground, Arena Mobile), data management (Collect, Arena), and geospatial analysis (SEPAL).[27]
  - **Weaknesses:** Its nature as a suite of separate tools makes it unsuitable as a unified backend to build upon directly. Its data models are highly complex, designed for large-scale national forest inventories, which may be overkill for Arboracle's initial needs.[28]

The following table summarizes this comparative analysis, clarifying the strategic fit of each platform.

Criteria	GeoNature	OpenTreeMap	Open Foris
Core Mission	Comprehensive Biodiversity Management	Crowdsourced Urban Tree Inventory	National Forest Monitoring & Analysis
Backend Tech	Python/Flask	Python/Django (Legacy)	Java, R, various
Database	PostgreSQL / PostGIS	Varies	PostgreSQL
API Style	REST / GeoJSON	Custom REST (not general purpose)	Various per tool
Community Health	High / Active	Low / Dormant	High / Institutional
Data Model	Comprehensive & Standardized	Tree/Plot Centric	Complex, Survey-based

Best Fit for Arboracle	**Conceptual Foundation & API Blueprint**	Feature Inspiration (Ecosystem Services)	Specialized Service Integration

The Recommended Path: A Hybrid, Service-Oriented Architecture

The analysis indicates that simply adopting a single open-source project in its entirety would be suboptimal. OpenTreeMap's technology is too dated and risky, while GeoNature's frontend is misaligned with modern best practices for user experience development.

Therefore, the most robust and strategic path forward is a hybrid, decoupled architecture. This approach intelligently combines the proven strengths of existing systems with the agility of a modern, bespoke technology stack.

1. **Adopt GeoNature's Data Model and API Philosophy as the Backend Blueprint:** The most valuable asset from these projects is GeoNature's mature, field-tested database schema and API design.[20] By using this as the conceptual foundation for Arboracle's backend, the project de-risks the most complex component—the data layer—and inherits a decade of scientific and community-driven refinement.[12] This ensures the application's data structure is scientifically sound and built for biodiversity from its inception.
2. **Build a Bespoke Frontend with Next.js and TypeScript:** A completely new frontend will be developed using modern web technologies. This provides full control over the user experience, enabling the implementation of the biophilic and biomimetic design philosophy. A modern Next.js/React codebase is also far more attractive to top-tier development talent than legacy frameworks.[31]
3. **Integrate Specialized Tools as Microservices:** Powerful capabilities, such as the AI-driven tree detection inspired by TreeTect or specific data processing modules from the Open Foris suite, will be treated as independent microservices.[13] This service-oriented architecture keeps the core application clean, resilient, and scalable. Individual services can be updated, replaced, or scaled independently without affecting the main application.

This decoupled strategy is the gold standard for modern application development. It demonstrates a sophisticated architectural plan that leverages the stability of proven models while embracing the innovation of new technologies, a combination that is highly appealing to investors and essential for long-term success.

The Arboracle Technical Blueprint

This section provides a granular, opinionated, and justified breakdown of the entire technology stack. It is intended to serve as a core technical manual for the development team, ensuring that Arboracle is built on a foundation that is modern, scalable, and secure.

Core Application Architecture: The Next.js/TypeScript Frontend

The frontend will be a state-of-the-art web application, designed for performance, developer experience, and a rich user interface.

- **Framework and Language:** Next.js 15 with TypeScript is the recommended choice. Next.js is the leading framework for building production-grade React applications, offering critical features like Server Components, Server Actions for mutations, a file-system-based App Router, and automatic image optimization.[33] TypeScript adds static typing, which is essential for building robust, maintainable applications at scale.[32]
- **Project Structure:** A professional, scalable project structure will be implemented from day one, following industry best practices.[31]
  - src/app/: The heart of the application, containing all routes, layouts, pages, and route groups as defined by the Next.js App Router conventions.[31]
  - src/components/: This directory will be organized to separate concerns clearly:
    - ui/: For primitive, reusable UI components like buttons, modals, and cards.[31]
    - layout/: For structural components like the main header, footer, and sidebars.[31]
    - features/: For complex components tied to specific business logic, such as TreeDataForm or EcosystemServicesChart.[31]
  - src/lib/: For modules that interact with external services, such as API clients, database connection helpers, and authentication logic.[31]

    - src/utils/: For pure, side-effect-free utility functions like date formatters or input validators.[31]
    - src/store/: For lightweight client-side state management using Zustand, with stores organized by domain (e.g., user.store.ts, map.store.ts).[31]
    - src/models/: For all TypeScript interfaces and type definitions that describe the application's core data entities (e.g., Tree, Observation, User).[31]
- **Styling:** Tailwind CSS will be used as the primary styling solution. Its utility-first approach allows for rapid development and makes it straightforward to implement the custom, nature-inspired design system without being constrained by a pre-built component library.[34]
- **State Management:** For client-side global state, Zustand is recommended for its simplicity and minimal boilerplate.[31] For managing server-side state, data fetching, and mutations, the application will leverage the powerful, built-in capabilities of Next.js Server Components and Server Actions, reducing the need for complex client-side data-fetching libraries.[34]

Core Application Architecture: The Backend API Service

The backend will be a dedicated API service, optimized for performance and seamless integration with AI and geospatial tools.

- **Language and Framework:** Python with the FastAPI framework. Python is the lingua franca of artificial intelligence and machine learning, making it the natural choice for a backend that needs to integrate with tools like PyTorch and geospatial libraries.[13] FastAPI is a modern, high-performance web framework that offers automatic interactive API documentation (via OpenAPI), data validation with Pydantic, and native asynchronous support, making it ideal for an I/O-heavy, database-driven application. It represents a significant performance and developer-experience upgrade over the Flask or Django frameworks used in older projects.[18]
- **API Design:** The API will be designed following RESTful principles and will primarily serve data in GeoJSON format. GeoJSON is the standard for encoding geographic data structures and is the format used by GeoNature's robust API.[20] Endpoints will be structured logically around core resources like /trees, /observations, /users, and /maps.

Core Application Architecture: The Database

The database is the long-term store of value for Arboracle and must be robust, scalable, and geospatially enabled.

- **System and Extension:** PostgreSQL (version 15 or higher) is the recommended relational database system due to its proven reliability, feature set, and strong open-source community. The PostGIS extension is non-negotiable; it transforms PostgreSQL into a powerful geospatial database capable of storing, querying, and analyzing location-based data efficiently. This combination is a core component of the GeoNature stack and is considered the industry standard for this type of application.[18]
- **Initial Data Model:** The MLP data model will be a pragmatic synthesis of the concepts from OpenTreeMap and GeoNature.
  - **From OpenTreeMap:** The model will adopt the clear distinction between a Plot (a physical location represented by a PostGIS geometry point) and a Tree (the biological entity at that location). This allows a single plot to have a history of different trees over time.[23]
  - **From GeoNature:** The model will incorporate a more comprehensive and extensible Observation entity. This table will link a user to a plot at a specific time and can contain notes, photos, and other data. This structure is designed to be extensible for future data types beyond trees, such as soil samples or fauna sightings. The architecture will also plan for tables to manage taxonomies and standardized nomenclatures, inspired by GeoNature's TaxHub and Nomenclatures modules.[18]
  - **Key MLP Tables:** users, plots (with a PostGIS geometry column), trees (with fields for species, diameter at breast height (DBH), height, etc.), observations (linking users, plots, and timestamps, with fields for notes and photo URLs), and species_taxonomy.

The AI & Geospatial Services Layer: A Serverless Approach

Architecting the AI and machine learning features as independent, scalable microservices is a critical strategic decision. This is particularly important for a startup, as it optimizes for cost, scalability, and resilience.

- **Tree Detection Service:** This service, based on the principles of TreeTect, will be the primary AI feature in the MLP.
    - **Technology:** The service will be a Python function built using a deep learning framework like PyTorch and potentially starting with an open-source library like detectree as a baseline.[13] Critically, it will be deployed as a serverless function (e.g., AWS Lambda) or on a managed container platform (e.g., AWS Fargate). This architectural pattern is directly informed by the design of TreeTect, which was explicitly created to run on AWS Lambda.[13]
    - **Workflow:** The user-facing application will upload an image to a cloud storage bucket (e.g., Amazon S3). This upload event will automatically trigger the serverless function. The function will process the image, execute the tree detection model, and write the results (e.g., a GeoJSON polygon of the canopy and a calculated health score) back to the PostgreSQL database, linking it to the original user observation.
- **Strategic Advantage of Serverless AI:** Choosing a serverless, event-driven architecture for the AI layer provides immense benefits. It avoids creating a monolithic backend where a long-running, computationally intensive task could block the entire API. Instead, it offers automatic scaling to meet demand, a pay-for-what-you-use cost model that is ideal for an early-stage venture, and improved resilience, as a failure in one AI service will not crash the core application. This modern, cloud-native approach is financially prudent and demonstrates a high level of architectural maturity to potential investors.

Mobile Strategy: PWA First

For the initial launch and MLP phase, the most efficient and effective mobile strategy is to build the Next.js application as a Progressive Web App (PWA).

- **Justification:** A PWA can provide a near-native experience on mobile devices, including features like offline access for data collection and an icon on the user's home screen. This is achieved without the significant time, cost, and complexity of developing and maintaining separate native codebases for iOS and Android. This approach allows the entire development team to focus its efforts on a single, unified codebase, accelerating the path to market. This strategy learns from the experience of projects like OpenTreeMap, whose separate native apps became outdated over time.[24] Dedicated native applications, such as those offered by

Open Foris Ground [11], can be explored as a post-investment growth strategy once the core product has found market fit.

Implementation Roadmap and Software Catalogue

This section provides the concrete, actionable steps and a definitive list of tools required to build the Arboracle MLP. It serves as a practical guide for the development team to execute the vision and technical blueprint.

Phased MLP Rollout Plan

A phased, 12-week plan is proposed to deliver the MLP in a structured and agile manner.

- **Phase 1: Core Infrastructure & Data Foundation (Weeks 1-4)**
  - **Objectives:** Establish the development environment and foundational backend services.
  - **Key Tasks:**
    - Set up the GitHub repository following best practices, including README.md, LICENSE, and CONTRIBUTING.md files.[39]
    - Configure a CI/CD pipeline using GitHub Actions for automated testing and deployment.[41]
    - Provision the PostgreSQL database with the PostGIS extension.
    - Develop the initial version of the Python/FastAPI backend, including the core data models and user authentication endpoints.
    - Bootstrap the Next.js frontend application with the defined project structure and styling configuration.
- **Phase 2: The "Digital Ecologist" MLP (Weeks 5-10)**
  - **Objectives:** Build the core user-facing features of the Minimum Lovable Product.
  - **Key Tasks:**
    - Implement the map-centric UI for data collection.
    - Develop the "Living Map" visualization feature, incorporating biophilic design elements.

        - Integrate the first version of the serverless TreeTect AI service for image-based identification.
        - Implement the ecosystem services calculation feature and display the results to the user.
        - Finalize and test the PWA configuration for a robust offline and mobile experience.
- **Phase 3: Community & Investment Readiness (Weeks 11-12)**
    - **Objectives:** Refine the product based on initial feedback and prepare for launch and investor conversations.
    - **Key Tasks:**
        - Implement the simple sharing feature to allow users to share their maps.
        - Conduct focused user testing with a small cohort of early adopters to gather feedback.
        - Iterate on the UI/UX based on testing results to maximize "lovability."
        - Prepare comprehensive product documentation and a compelling demo for investor pitches.

Comprehensive Software and Repository Catalogue

This catalogue provides a vetted, unambiguous list of the software, libraries, and platforms recommended for building Arboracle. It is designed to eliminate research time and ensure the development team starts with a best-in-class, modern toolchain.

Category	Software/Library	Role in Arboracle Stack	Repository / Link
Frontend Framework	Next.js	Core web application framework for UI, routing, and server-side rendering.	github.com/vercel/next.js
Frontend Language	TypeScript	Provides static typing for robust, scalable, and maintainable code.	(https://github.com/microsoft/TypeScript)
UI Components	Shadcn/ui	A collection of beautifully designed,	github.com/shadcn-ui/ui

		accessible, and unstyled components.	
Styling	Tailwind CSS	Utility-first CSS framework for rapid UI development.	github.com/tailwindlabs/tailwindcss
Client State	Zustand	Lightweight global state management for React.	github.com/pmndrs/zustand
Mapping Library	Mapbox GL JS / React Map GL	High-performance, interactive map rendering.	github.com/visgl/react-map-gl
Backend Framework	FastAPI	High-performance Python framework for building the core API.	github.com/tiangolo/fastapi
Database	PostgreSQL	Primary relational database for all application data.	www.postgresql.org
DB Extension	PostGIS	Geospatial extension for PostgreSQL to handle location data.	postgis.net
AI/ML Foundation	PyTorch	Core deep learning framework for building/running tree detection models.	github.com/pytorch/pytorch
Tree Detection Ref.	detectree	Python library for tree segmentation from aerial imagery. A starting point.	github.com/martibosch/detectree
Containerization	Docker	For creating consistent development and production environments.	github.com/docker/cli
CI/CD	GitHub Actions	Automating builds, tests, and deployments directly	docs.github.com/en/actions

		from the repository.	
Code Quality	ESLint	Pluggable linter for identifying and fixing problems in JavaScript/TypeScript.	github.com/eslint/eslint
Code Formatting	Prettier	Opinionated code formatter to ensure consistent style across the codebase.	github.com/prettier/prettier
Conceptual Ref.	GeoNature	Open-source project providing the foundational data model and API concepts.	(https://github.com/PnX-SI/GeoNature)
Conceptual Ref.	Open Foris	Suite of tools for inspiration on specific data collection workflows (e.g., Ground).	github.com/openforis

Conclusion and Recommendations

This report outlines a comprehensive and strategic plan for the development of Arboracle. The recommendations are designed to produce a Minimum Lovable Product that is not only technically sound and ready for investment but also philosophically coherent and deeply aligned with the values of biodiversity and nature.

The key recommendations are:

1. **Embrace a "Nature-Principled" Brand:** Move beyond simple aesthetics by integrating biophilic and biomimetic principles into the core UI/UX and functionality. This creates a deeply authentic and defensible brand identity.
2. **Adopt a Hybrid, Decoupled Architecture:** Leverage the proven, scientifically robust data models from the GeoNature project as a blueprint for a new Python/FastAPI backend. Build a modern, bespoke frontend using Next.js/TypeScript to ensure a world-class user experience and attract top talent.

3. **Utilize a Serverless AI Layer:** Architect AI features like tree detection as independent, event-driven microservices. This approach is cost-effective, highly scalable, and resilient—key attributes for a successful startup.
4. **Pursue a PWA-First Mobile Strategy:** Deliver a high-quality mobile experience efficiently by focusing all development resources on a single, unified codebase, deferring the cost and complexity of native apps until after market validation.

By following this blueprint, the Arboracle team can confidently build a product that is both lovable to its first users and architected for scalable, long-term growth. The plan balances innovation with pragmatism, ensuring that resources are focused on delivering maximum value and creating a compelling case for future investment.

Works cited

1. Nature in Digital Design: Biophilic Design in UI/UX - Code Like A Girl, accessed July 4, 2025, https://code.likeagirl.io/nature-in-digital-design-biophilic-design-in-ui-ux-36c0dd80a54d
2. Nature Remastered - Pantone, accessed July 4, 2025, https://www.pantone.com/articles/color-palettes/nature-remastered
3. 5 Lessons You Can Learn from Nature as UI/UX Designers, accessed July 4, 2025, https://lemonyellow.design/blog/design/5-lessons-you-can-learn-from-nature-as-uiux-designers/
4. Natural color palettes for UI design | by Anna Grand - UX Planet, accessed July 4, 2025, https://uxplanet.org/introduction-to-natural-palettes-9503bfeee3d5
5. How to Find Graphic Design Inspiration From Biomimicry and Natural Selection, accessed July 4, 2025, https://bl3nddesign.ca/biomimicry-and-natural-selection
6. Nature Inspired designs, themes, templates and downloadable graphic elements on Dribbble, accessed July 4, 2025, https://dribbble.com/tags/nature-inspired
7. Nature UI designs, themes, templates and downloadable graphic elements on Dribbble, accessed July 4, 2025, https://dribbble.com/tags/nature-ui
8. Nature Web Design - Pinterest, accessed July 4, 2025, https://www.pinterest.com/ideas/nature-web-design/928569471060/
9. The Future of Biomimicry in Web Design: Drawing Inspiration from Nature - Acodez, accessed July 4, 2025, https://acodez.in/biomimicry-in-web-design/
10. Branding Toolkit - Convention on Biological Diversity, accessed July 4, 2025, https://www.cbd.int/gbf/branding
11. Open Foris Ground - Apps on Google Play, accessed July 4, 2025, https://play.google.com/store/apps/details?id=org.openforis.ground
12. GeoNature - Natural solutions, accessed July 4, 2025, https://www.natural-solutions.world/geonature
13. krakchris/TreeTect: Tree detection - GitHub, accessed July 4, 2025, https://github.com/krakchris/TreeTect
14. TreeTect | Boston.gov, accessed July 4, 2025,

https://www.boston.gov/departments/new-urban-mechanics/treetect
15. Accuracy of a LiDAR-Based Individual Tree Detection and Attribute Measurement Algorithm Developed to Inform Forest Products Supply Chain and Resource Management, accessed July 4, 2025, https://northwestmanagement.com/wp-content/uploads/2022/01/AMAD-process.pdf
16. OpenTreeMap Open Source - Mapping all the trees of the world, one at a time!, accessed July 4, 2025, https://opentreemap.github.io/
17. PyBulls/OTM2: OpenTreeMap is a collaborative platform for crowdsourced tree inventory, ecosystem services calculations, urban forestry analysis, and community engagement. - GitHub, accessed July 4, 2025, https://github.com/PyBulls/OTM2
18. Installer GeoNature - Réserves Naturelles de France, accessed July 4, 2025, https://si-en-reseau.reserves-naturelles.org/documentation-geonature/installation.html
19. (PDF) GeoNature, Open-Source FAIR Biodiversity Data Management - ResearchGate, accessed July 4, 2025, https://www.researchgate.net/publication/354869690_GeoNature_Open-Source_FAIR_Biodiversity_Data_Management
20. Geonature v2 - Postman, accessed July 4, 2025, https://documenter.getpostman.com/view/2640883/RWaPskTw
21. GeoNature - Cat OPIDoR, accessed July 4, 2025, https://cat.opidor.fr/index.php/GeoNature
22. GEONATURE, UN OUTIL OPEN SOURCE DÉVELOPPÉ PAR LES PARCS NATIONAUX FRANÇAIS, accessed July 4, 2025, https://geonature.fr/documents/2018-01-GeoNature.pdf
23. otm-core/doc/api.md at develop - GitHub, accessed July 4, 2025, https://github.com/OpenTreeMap/otm-core/blob/develop/doc/api.md
24. OpenTreeMap - GitHub, accessed July 4, 2025, https://github.com/OpenTreeMap
25. github.com/opentreemap/otm-ecoservice | Go | Open Source Insights, accessed July 4, 2025, https://deps.dev/go/github.com%2Fopentreemap%2Fotm-ecoservice/v0.0.0-20230502152157-043deffec7d9
26. Open Foris: Free open-source solutions for forest and land monitoring - Food and Agriculture Organization of the United Nations, accessed July 4, 2025, https://www.fao.org/in-action/openforis/overview/en
27. Open Foris, accessed July 4, 2025, https://openforis.org/
28. OPEN FORIS ARENA — Openforis Arena documentation, accessed July 4, 2025, http://arena-dguerrero.readthedocs.io/
29. Materials - Open Foris, accessed July 4, 2025, https://openforis.org/materials/
30. Wetland management with GeoNature - Natural solutions, accessed July 4, 2025, https://www.natural-solutions.world/blog/wetlandmanagement
31. The Ultimate Guide to Organizing Your Next.js 15 Project Structure - Wisp CMS, accessed July 4, 2025, https://www.wisp.blog/blog/the-ultimate-guide-to-organizing-your-nextjs-15-pro

ject-structure
32. Next.js and TypeScript Component Organization 101: Best Practices - Fishtank, accessed July 4, 2025, https://www.getfishtank.com/insights/best-practices-for-nextjs-and-typescript-component-organization
33. Getting Started: Project Structure | Next.js, accessed July 4, 2025, https://nextjs.org/docs/app/getting-started/project-structure
34. Getting Started: Project Structure - Next.js, accessed July 4, 2025, https://nextjs.org/docs/13/getting-started/project-structure
35. Getting Started: Project Structure | Next.js, accessed July 4, 2025, https://nextjs.org/docs/pages/getting-started/project-structure
36. React/Next.js(Typescript) Project Architecture Setup: A Beginner's Guide - Medium, accessed July 4, 2025, https://medium.com/@TGod-Ajayi/react-next-js-tyepscript-project-architecture-setup-a-beginners-guide-a63d8469805b
37. martibosch/detectree: Tree detection from aerial imagery in Python - GitHub, accessed July 4, 2025, https://github.com/martibosch/detectree
38. AI-enabled tree inventory | Connecting Nature, accessed July 4, 2025, https://connectingnature.eu/cnep-opportunities/ai-enabled-tree-inventory
39. Best practices for repositories - GitHub Docs, accessed July 4, 2025, https://docs.github.com/en/repositories/creating-and-managing-repositories/best-practices-for-repositories
40. GitHub Repository Structure Best Practices | by Soulaiman Ghanem | Code Factory Berlin, accessed July 4, 2025, https://medium.com/code-factory-berlin/github-repository-structure-best-practices-248e6effc405
41. GeoNature/geonature.github.io - GitHub, accessed July 4, 2025, https://github.com/GeoNature/geonature.github.io