Project Proposal

Corbyn Robinson

Open Window, School of Fundamentals

Interactive Development DV200

Tsungai Katsuro

24 July 2025

Table of Contents

Client Conceptualisation & Problem Statement

Project Overview

Poseidon's Notebook is a web-based marine life observation log designed for:

- Scuba divers to record sightings
- Marine biology students for research tracking
- Ocean enthusiasts to maintain personal logs

The business domain is environmental conservation and marine

Core Problem being Solved

- No centralized, simple tool for casual marine life logging
- Existing solutions are either:
    - Too complex (research-grade databases)
    - Too limited (paper notes, generic note-taking apps)
- Data fragmentation – observations often lost across multiple platforms

Why a Software Solution is Necessary

Current Problem	**Poseidon's Notebook's solution**
Paper notes get lost/wet	Digital preservation
No species reference	Built-in common species database
Can't analyze trends	Filterable logs with date/species/location

Constraints and Limitations

Constraint	**Poseidon's Notebook's solution**
Limited to common species	Pre-populate 50 most common marine species
No image processing	Use placeholder icons for species
Single-user focus	Simple JWT auth (no social login required)

Brand Development

Name: Poseidon’s Notebook

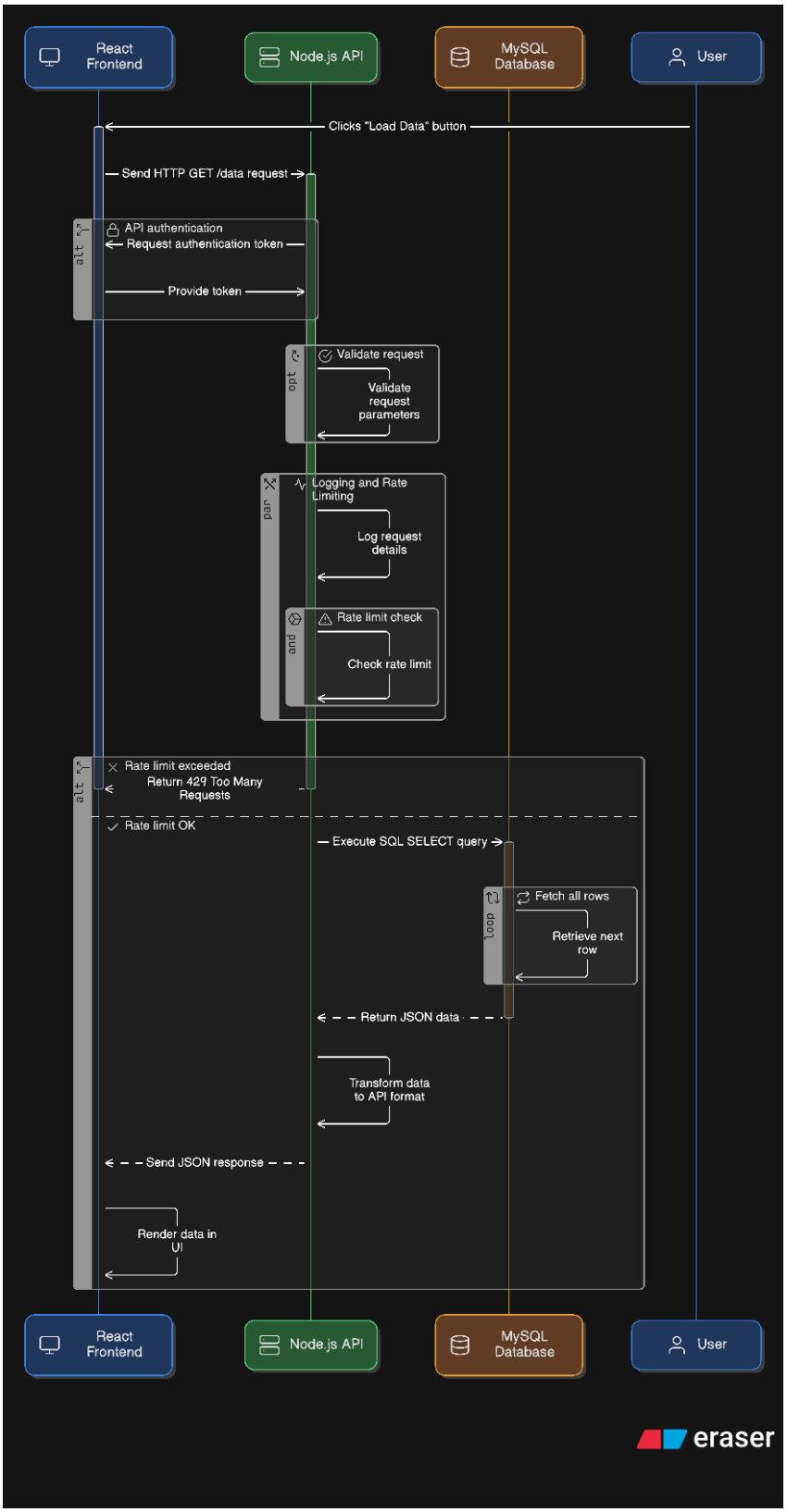
Tagline: "Log. Learn. Protect."

Logo Concept (subject to change):

- Wave silhouette + fish icon
- Color palette: #2A9D8F (teal), #264653 (dark blue), #E9C46A (accent yellow)
- Font Pairing:
- Headings: Montserrat Bold
- Body: Open Sans Regular

System Architecture

High-leveled System Diagram



Technology Stack Justification

Component	Technology	Reasoning
Frontend	React	Component reusability and strong statement management
Styling	CSS Modules	Scoped styling with no conflicts
Backend	Node.js and Express	Lightweight
Auth	JWT	Simple implementation for single-user app

Feature Requirements

Scope Definition

Included in the minimum viable project (MVP):

- User authentication (register/login)
- CRUD operations for sightings:
    - Create: Log new observation
    - Read: View personal logs
    - Update: Edit existing entries
    - Delete: Remove observations
- Species reference database
- Basic filtering (by species, date, location)

Explicitly Excluded (Version 1):

- Image uploads
- GPS/map integration
- Multi-user collaboration

SMART Objectives

By Week 7: Functional backend API with MySQL connection

By Week 8: Complete React frontend with basic CRUD

By Week 10: Implement JWT authentication

By Week 13: Add filtering and basic data visualization

By Week 15: Deployment

Feature Prioritisation

MVP:

1. USer authentication
2. Sightings CRUD:
    a. Date, location, species dropdown
    b. Notes field (text area)
3. Pre-populated species database (50 entries)

Nice to have:

1. Simple charts (sightings per month)
2. Export to CSV functionality
3. Dark mode toggle

Future Considerations

1. Photo uploads
2. Community sharing features
3. Mobile app version

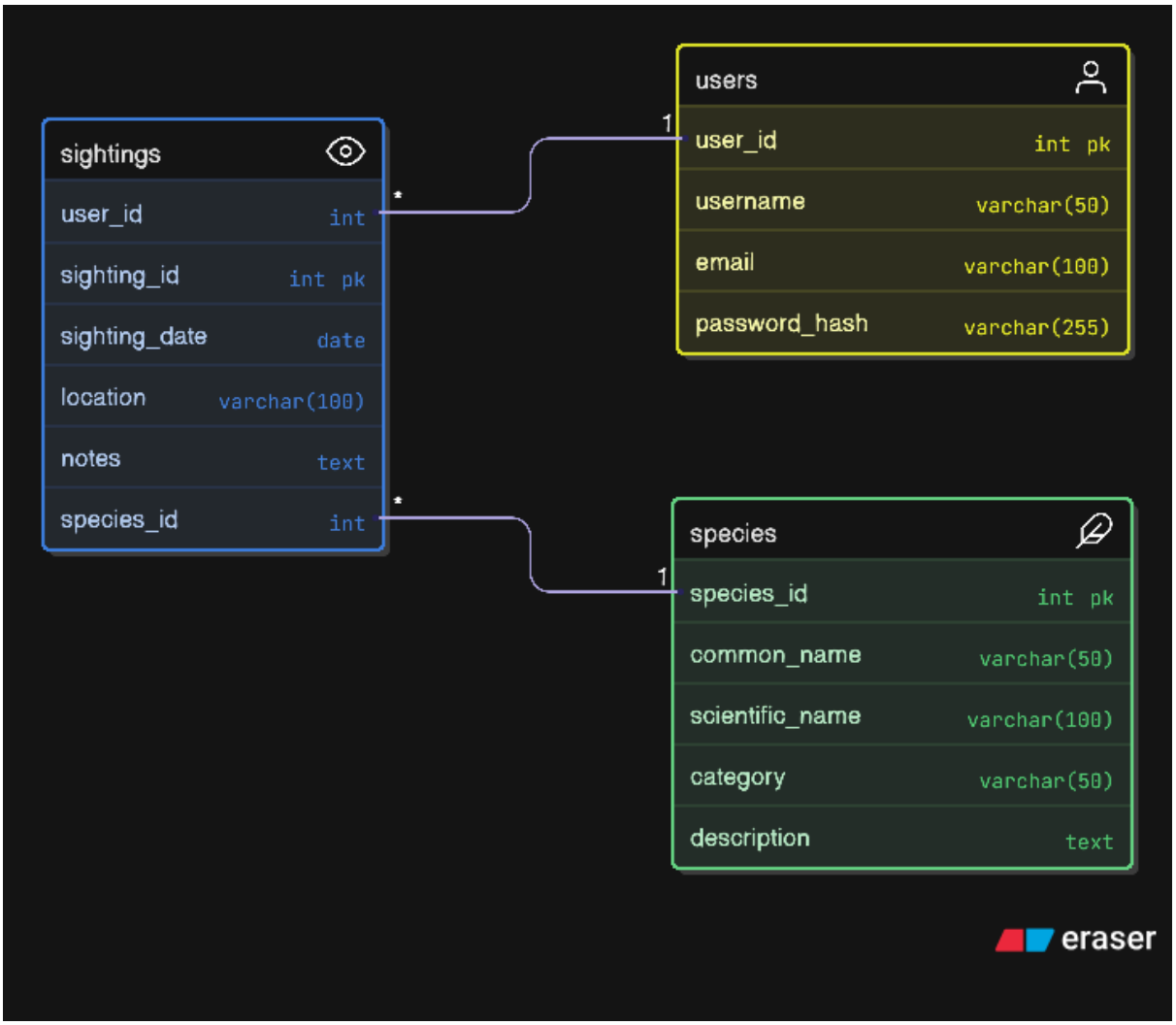
User Roles and Permissions

Role	**Permissions**
Guest	View public species database

Logged-in User	Full CRUD on own sightings
Admin	Manage species database

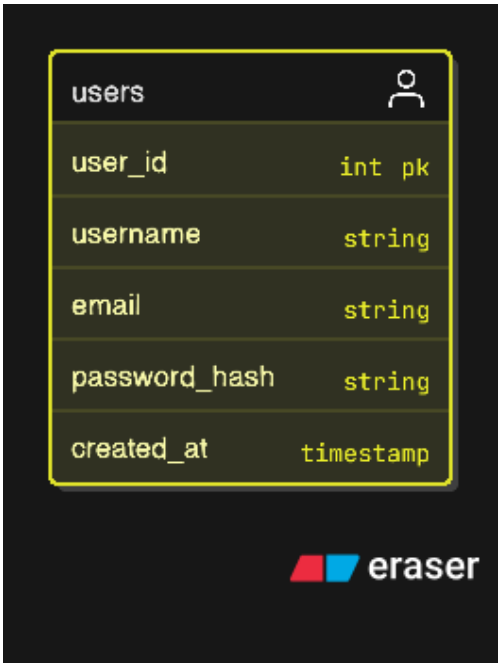
Data Planning

Entity-Relationship Diagram



Database Schema Details

Users Table



Relationships:

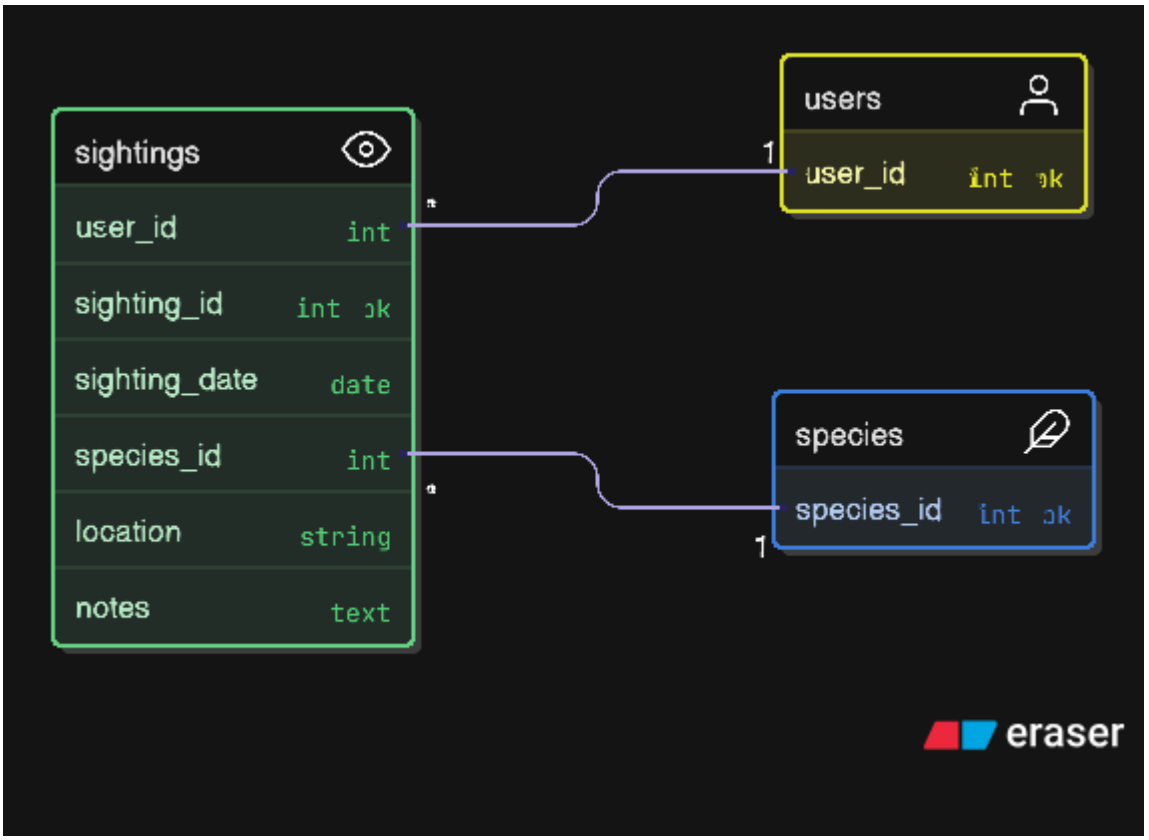
- One-to-Many with sightings (1 user → many sightings)

Species Table (Pre-populated):

Relationships:

- One-to-Many with sightings (1 species → many sightings)

Sightings Table:

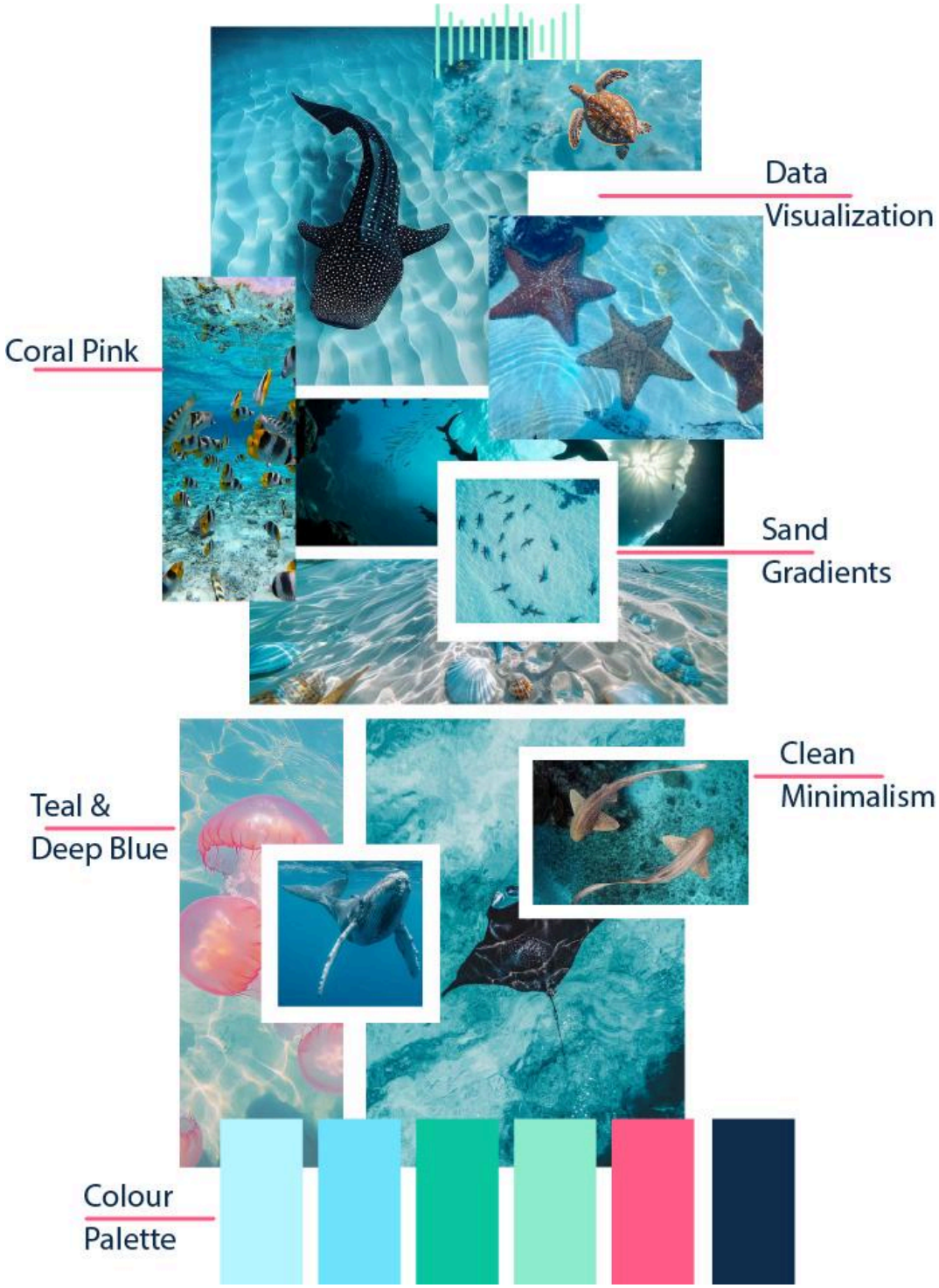


Key Constraints

- Referential Integrity:
  - sightings.user_id → users.user_id (CASCADE delete)
  - sightings.species_id → species.species_id (RESTRICT delete to prevent orphaned sightings)
- Data Validation:
  - sighting_date cannot be future dates (enforced via application logic)
  - depth_meters range: 0.00 to 200.00 (enforced via CHECK constraint if MySQL 8.0+)

Wireframes and UI/UX Considerations

Moodboard and Design System



Colour palette:

- Primary Colors (Dominant Brand Colors)
    - #0F2D4C (Deep Navy) → Headers, primary buttons, footer

    - #0AC7A1 (Vibrant Teal) → Main CTAs, interactive elements
- Secondary Colors (Supporting Colors)
    - #6EE4FF (Sky Blue) → Secondary buttons, info cards
    - #8EEDCD (Mint Green) → Accent backgrounds, success states
- Accent Colors (Highlights & Alerts)
    - #FF5C87 (Coral Pink) → Error messages, important alerts
    - #B6F6FF (Pale Aqua) → Hover states, subtle accents
- Neutrals (Structure & Text)
    - #FFFFFF (White) → Backgrounds, card surfaces
    - #000000 (Black) → Body text, dark UI elements
    - #F5F5F5 (Light Gray) → Secondary backgrounds (optional)

UI Inspiration:

- Clean, card-based layout
- Underwater imagery as subtle backgrounds
- Rounded corners for friendly feel

Wireframes

(Will be done next week)

Accessibility Considerations

- Color Contrast: AA compliant (4.5:1 minimum)
- Keyboard Navigation: All forms operable via keyboard
- ARIA Labels: For screen readers
- Font Size: Minimum 16px for body text

Project Timeline and Workflow

Development Phases

Phase	Weeks	Deliverables
Setup	3 - 4	Tech stack setup, DB design
Backend	5 - 8	API endpoints, auth system
Frontend	9 - 11	Core pages, CRUD functionality
Features	12 - 13	Filtering, basic charts
Polish	14 - 15	UI refinements, error handling
Deployment	16	Live deployment, final testing

Weekly Breakdown

- Week 3-4:
    - Set up React app
    - Install Express.js
    - Design and create MySQL database
- Week 5-8:
    - Build API endpoints:
    - POST /api/sightings (create)
    - GET /api/sightings (read all)
    - GET /api/species (reference)
    - Implement JWT authentication
- Week 9-11:
    - React components:
    - SightingsList
    - SightingForm
    - SpeciesBrowser

    - Connect frontend to API
- Week 12-13:
    - Add filtering functionality
    - Implement simple charts (Chart.js)
    - Basic error handling
- Week 14-15:
    - UI polish (loading states, empty states)
    - Responsive design testing
- Week 16:
    - Deployment
    - Final documentation

Project Management Approach

Methodology: Agile (Kanban)

Tools:

- Trello Board:
    - Columns: Backlog, In Progress, Testing, Done
    - Cards for each feature with checklists
- GitHub:
    - Feature branches
    - Semantic commit messages
    - Weekly merges to main

Risk Management:

- Backend delays: Focus on API-first development
- UI challenges: Use component libraries if needed
- Scope creep: Stick strictly to MVP features

Risks, Challenges & Conclusion

Technical Risks

Risk	Likelihood	Impact	Mitigation Strategy
API connection failures	Medium	High	Implement robust error handling
Database performance	Low	Medium	Optimise queries, add indexes
Authentication issues	Medium	High	Thoroughly test auth flow

Non-Technical Risks

Challenge	Solution
Time management	Strict weekly milestones
Design skills	Use UI component libraries
Deployment issues	Early staging deployment

Why this Project will Succeed

- Focused Scope: Limited to core CRUD operations
- Educational Value: Great for learning full-stack development
- Real-World Application: Solves actual user needs
- Manageable Complexity: Appropriate for DV200 level

Final Thoughts

Poseidon's Notebook provides a perfect balance of:

- Practical functionality
- Technical learning opportunities
- Creative design potential

Approval Request: This proposal demonstrates thorough planning with achievable milestones. I welcome any feedback to refine the approach before development begins.