**1. Project Objectives**

The goal of this project is to develop an interactive web dashboard that visualizes Harmful Algal Bloom (HAB) data and integrates a Large Language Model (LLM) such as GPT-4 to provide human-readable explanations for bloom alerts. HABs are a growing threat to marine ecosystems, aquaculture operations, and public health, yet interpreting bloom predictions remains challenging for many stakeholders due to the technical nature of the data.

This project addresses that gap by combining dynamic visualizations with natural language explanations. The system will allow users—such as aquaculture managers and environmental researchers—to interact with time-series data (e.g., Chlorophyll-a levels, sea surface temperature) and spatial bloom alerts through an intuitive dashboard interface. The integration of an LLM will enable users to ask questions such as “Why is this site at high risk today?” and receive detailed explanations of contributing factors (e.g., rapid temperature rise or turbidity changes), along with possible mitigation steps.

The core deliverables of the project include:

* Interactive visualizations for time-series trends and geospatial risk mapping
* A conversational chat interface powered by an LLM for interpretability
* Options to download visual data (as PNG, JPG, etc.)
* A fully containerized frontend (React) and backend (Flask/Express) deployed using Docker

Ultimately, this dashboard aims to enhance the transparency, usability, and actionability of HAB prediction models. By making complex environmental data more understandable and accessible, the system supports informed, real-time decision-making for sustainable aquaculture and ecosystem management.

**2. Sprint Plan**The project will follow agile development principles, divided into weekly sprints aligned with module deadlines.   
  
Week 1–2: Team formation, project scoping, and data source research  
Week 3: Initial UI mockups and backend API structure draft  
Week 4: Build time-series and map visualizations using dummy data  
Week 5: Integrate LLM interface and prompt engineering  
Week 6: MVP completion, usability test with peers, and interim presentation prep  
Week 7–8: Refine UI/UX, handle real-time data slices, add download options  
Week 9: Technical and user evaluation setup and testing  
Week 10: Final presentation and demo prep  
Week 11–12: Report writing, fine-tuning, deployment and submission

Sprint tracking will be done via Trello and GitHub

**3. Roles (TBD)**

* **Project Manager:** Coordinates meetings, task allocation, and sprint tracking
* **Frontend Developer(s):** Implements dashboard UI and visualizations (React, Plotly.js, Leaflet)
* **Backend Developer(s):** Develops API routes and LLM interface (Flask/Express)
* **LLM Integration Lead:** Designs prompt templates, handles API calls, manages explanation quality
* **Data Engineer:** Handles data sourcing, preprocessing, and structuring for dashboard use
* **DevOps/Deployment:** Manages Docker setup, CI/CD via GitHub Actions, and cloud hosting

**4. Architectural Setup**

* **Frontend**: React app hosted on Vercel
* **Backend**: Flask API deployed on Render (handles LLM queries, data endpoints)
* **LLM**: OpenAI GPT-4 API with limited free credits and cost controls; fallback to Hugging Face-hosted LLMs
* **Data Storage**: Static CSVs for sensor data; MongoDB Atlas (free tier) for query logging
* **Visualization**: Plotly.js for time-series, Leaflet.js for spatial alerts
* **Containerization**: Docker for both frontend and backend
* **CI/CD**: GitHub + GitHub Actions for auto-deployment on push to main

**5. Data Plan**

* **Sources**:
  + HAEDAT: https://haedat.iode.org
  + HAIS: https://data.hais.ioc-unesco.org
  + Bloomin’ Algae: https://www.ceh.ac.uk/our-science/projects/bloomin-algae
* **Format**: CSV or JSON slices with Chl-a, SST, turbidity, and bloom labels
* **Processing**: Python scripts to clean, merge, and prepare data
* **Ethics**: All data is publicly available and used solely for educational/non-commercial purposes

**6. GitHub Usage**

* Central GitHub repo for all code, updated regularly
* Commit standards and branch workflows will be enforced
* Team members must push design docs, minutes, diagrams, code, and testing data continuously

**7. Team Management**

* **Meetings**: Twice-weekly standups + ad hoc pair programming
* **Tools**: Trello, GitHub Projects, Zoom/Google Meet
* **Minutes**: Shared via Google Docs and committed weekly to GitHub
* Issues tracked using Trello cards and GitHub Issues with labels and assignees

**8. Evaluation Plan**

* **Usability**: Peer testing with a timed task (e.g., ask for explanation at site A)
* **Performance**: Dashboard loads <2s; LLM response <5s
* **Accuracy**: LLM outputs must cite top 2 driving features per site, verified against known HAB drivers
* **Engagement**: Feedback will be collected to iteratively improve explanations

**9. Risk and Mitigation**

|  |  |  |
| --- | --- | --- |
| Risk | Impact | Mitigation |
| API costs (OpenAI) | Medium | Use free tokens for development, monitor usage, switch to Hugging Face if limits exceeded |
| Data access delays | Low | Use cached datasets and simulate live slices |
| LLM misinterpretation | Medium | Refine prompts; add fallback explanations |
| Team scheduling | Low | Fixed standup slots and async comms tools |