

Scuba Diving Log Web App 🐟

A client-side web application for scuba divers to log, edit, and review their dive experiences.

 [View the app online](#)

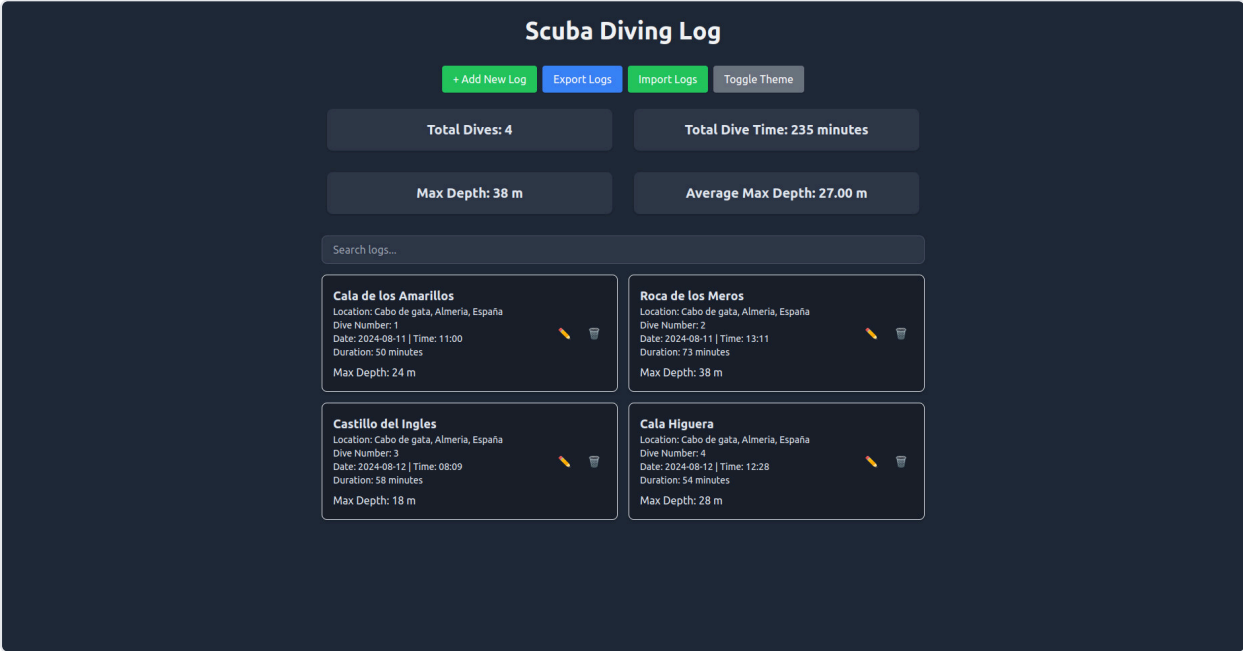
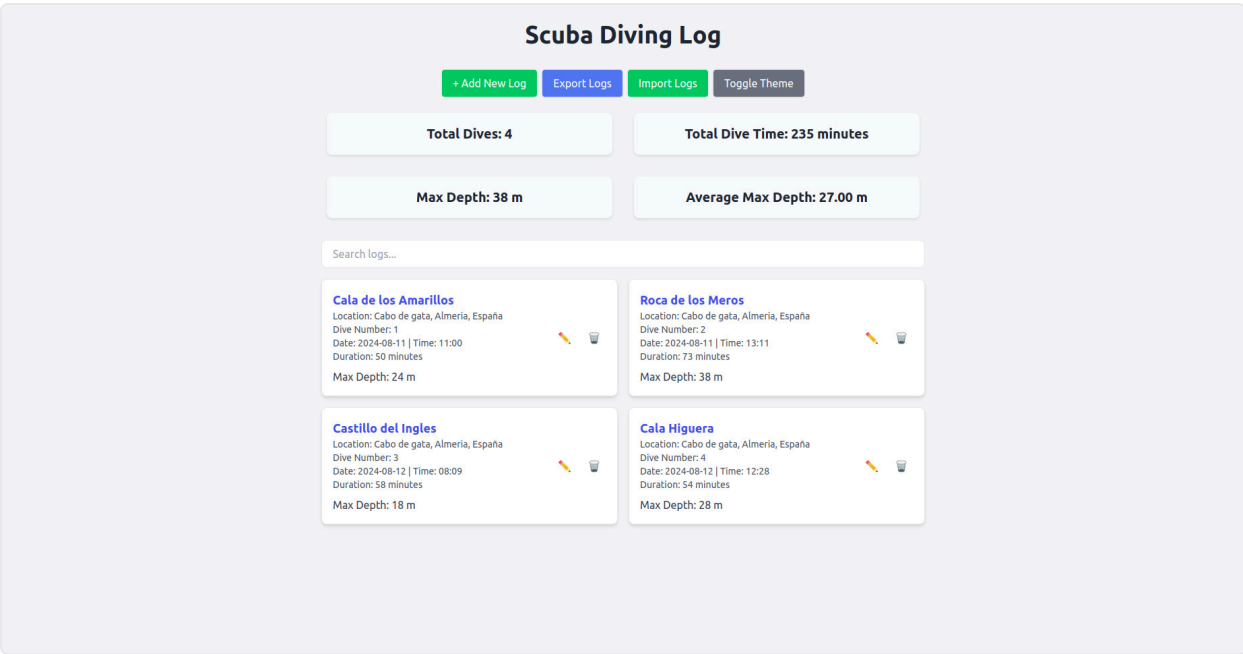
Introduction 📖

A dive log is a record of the diving history of an underwater diver. Recreational divers are generally advised to keep a logbook as a record, while professional divers may be legally obliged to maintain a logbook which is up to date and complete.

Still nowadays many divers choose to keep their logs in physical books since available digital solutions do not fulfil their needs. The proposed web application showcases a web-based diving log including the following features:




- **Create, Edit, and Delete Diving Logs:** Manage each dive experience with detailed information. 📝
- **Persistent Data Storage:** Dive logs are stored locally on the browser using LocalStorage. 💾
- **Display global stats:** Track global metrics across dives such as number of dives, total dive time, max depth... 📊
- **Search:** Search bar for easy log lookup. 🔍
- **Import/Export:** Import/export logs from/into JSON format. 📁
- **Dark/Light Mode Toggle:** Toggle between light and dark modes for comfortable viewing. ☀️🌙
- **Responsive UI:** An adaptive design using Tailwind CSS that looks great on both mobile and desktop. 📱💻

Below is a screenshot of the main interface, in both light and dark modes.



How to Use

To get started, open the application in a web browser and follow these steps:

- 1. Click on "Add New Log" to create a new dive log entry.
- 2. Fill out the details for each field, including dive location, date, depth, and other relevant information. 
- 3. To edit or delete a log, use the respective icons on each entry. 
- 4. All data is stored locally in your browser, so you can return to your logs any time from the same device. 

Development Process

This application was built using ChatGPT in about 17 iterations. Check out all stages on [GitHub](#)

The basics (iterations 1-3): Our first goal was to get a very basic but functional dive log. I took a couple of iterations to get there. Our initial prompt was:

```
Create a scuba diving log web application. The application MUST:
```



1. Allow the user to create, edit and delete diving logs.
2. Display the stored logs in a list.
3. Create a detail view for each log that opens up in a modal, that allows to edit and delete.
4. If there are no diving logs yet, it should have an appealing placeholder that inspires to add one.
5. Use an appealing, clean, minimalistic style with Tailwind CSS
6. Include all needed HTML, CSS and JavaScript code in a single file.
7. Load Tailwind CSS from a CDN
8. Operate completely in the client-side, and use LocalStore to store the data
9. Give clear instructions on how to open and use the application on a double-click.

```
Generate the complete code, and make sure that it can be executed just by opening the resulting HTML file.  
Ask me anything you need.
```

Feature enrichment (iterations 4-10): After getting to a basic functional state, we asked the IA for suggestions on how to improve the application and we went applying some of the suggestions sequentially.

Final tweaks and fixes (iterations 10-17): Final bugfixing, details and styling tweaks where we required some manual code inspection and manual amends at some points. Also here, we started requesting for partial code changes rather than complete code generation.

Challenges and learnings

- **Unattentive IA:** When new features were requested to be added, previous available features would often result broken and unavailable. 
- **Untested code:** A couple of times we had execution problems and had to provide console error logs to the IA to get them fixed. 
- **Manual intervention:** At some point, specifically with fine details, the IA would miss the capability to fix the problems that was requested to fix, requiring from human manual

intervention or at least manual inspection to better understand the problems source and therefore defining more specific and narrowed requests. 🔍

- **Progressively Lazy IA:** Specially on the last stages of the development, the IA would stop providing the complete application code, even when requested so. Therefore, at some point we stopped requesting the full code but only for partial changes. 🤖
- **Unstable and buggy ChatGPT platform:** Had to deal sometimes with "Oooops, there was a problem with your request" situations by ChatGPT platform which some have not been even resolved yet. ⚠️

Future work 🌐

- **UX:** Extend log data information.
- **UX:** Extend sorting and filtering options in log list.
- **UX:** Extend available general stats and main dashboard. 📊
- **UX:** Improve error management.
- **Backend:** Integrate with backend API for cloud data storage. ☁️
- **Style:** Improve general style and data visualization. 🎨