amplabs Documentation

Release latest

Apr 27, 2023

CONTENTS

Table of Contents

CHAPTER

ONE

ABOUT US

AmpLabs helps teams build better batteries.

Building off of the great Open Source work related to the Battery Data Genome, we are developing the components to realize an Open Source Battery Data Platform to help the entire battery value chain cut down on redundant work.

We want to close the gaps on solved problems so that teams can focus on building better batteries.

Try out the live hosted version of AmpLabs by visiting https://app.amplabs.ai

If you are interested in deploying AmpLabs in your own lab, you can do so with the Ampcloud Service

If you are interested in reviewing and giving feedback on the schema that AmpLabs natively runs, you can do so at AmpLabs Battery Datafile (BDF)

CHAPTER

TWO

PROBLEM STATEMENT

Inspired by Battery Lifecycle Framework with integrations into other community projects such as BEEP and Galvanalyser. AmpLabs puts these components together in a scalable and performant way to bring the concepts introduced by the Battery Data Genome in a user friendly manner.

2 Fundamental Problems facing the Battery Industry as Identified in the Principles of the Battery Data Genome

Heterogeneity

> Establishing the data and metadata conventions that will make heterogeneous data useful and enable interoperability, and rapid, large-scale capture of data from many sources and contributors

Scale

> Modern data science methods require large amounts of data and the battery community lacks the requisite scalable, standardized data hubs required for immediate use of these approaches. Lack of uniform data practices is a central barrier to the scale problem.

2.1 Project Obectives

Build useful software that **anyone** who works with batteries can use

Build a sturdy **foundation** upon which other developers can build on top

Build a library of **reusable components** that develop software & data standards

2.2 Key Areas of Focus

- Library of Cycler Converters
- Schema Mapping
- API Ecosystem
- Dashboard with Standard Plots

CHAPTER

THREE

COMPONENTS

How AmpLabs' Software is organized?

AmpLabs' Software Stack contains 5 conceptual components: Producer, Agent, Warehouse, Client, and Consumer.

The **Producer** creates the data. This would generally refer to something like a battery cycler or electric car that generates data about a battery.

The **Agent** is responsible for cleaning and collecting that data, and uploading it to the central data Warehouse. An Agent would be run by the same person running the Producer, and the **Warehouse** is the central data store that enables the rest of the data analysis and sharing.

A **Client** would be the application that gives a user easy access to their data. (It's worth nothing that in many cases, an Agent and Client may be a single package.) Finally the Consumer is the aforementioned end user, who is interested in exploring and analyzing the data.

The components for the Agent, Warehouse, and Client are made real in the ampcloud-service repo

More info for how the conceptual structure maps to specific code will be the focus for each of the following sections.

3.1 Producer

3.1.1 Producers

1. What cyclers are supported?
2. What columns are required from each type of cycler? What units are expected?

3.2 Agent

3.2.1 Agents

1. **How do I configure or tell the agent where to look for data**
    1. File based systems
    2. SQL based systems
2. What triggers a data collection event? is there a user defined schedule? Is the user supposed to kick off individual jobs?
3. How does the agent handle ongoing tests?

3.3 Warehouse

3.3.1 Warehouse

1. Where is my data stored?
2. Can I use my own instance of SQL (or whatever database technology the project uses).
3. Is the schema exposed? i.e. can I query the warehouse directly?

Since the goal of the "client" is to give users easy access to the data, we would like to build tools with which a user simply selects what data the user wants to plot, and then the tools output that plot.

I propose the following instructions.

For a new user to start, the new user should get Amplabs "user_token" and "cell_id" first.

3.4 Client

3.4.1 Client

Getting User Token and Cell ID

For a new user to start, the new user should get Amplabs "user_token" and "cell_id" first.

How to add "user_token" and "cell_id" to your code snippet.

1. Go to the amplabs.ai website, and click "try Amplabs Cloud".
2. Sign up and log into Amplabs Cloud. There, on the top right menu, click "get API token".
3. Copied the token into the "ipynb" file in Jupyter notebook or Google Colab to replace the "your-token" in the "user_token = 'your-token' " line.
4. In the Amplabs Cloud page that has been logged in, choose a sample at the left side of the webpage and click "Load" at the top left.
5. Copy the "Cell Id" and paste it into the "ipynb" file in Jupyter notebook or Google Colab to replace the "your-cell-id" in the "cell_id = 'your-cell-id'" line.

After getting Amplabs' "user_token" and "cell_id," using the Amplabs battery data visualization Jupyter notebook file is the next step.

Installing Jupyter Notebook

If you don't know what Jupyter Notebook is, it is a tool that makes Python easier to program. To install Jupyter notebook, there are 2 options below:

- Open a terminal, type "pip install jupyter". Once installed, type "jupyter notebook" in the terminal to open Jupyter notebook.
- Open a terminal, type "pip install jupyterlab". Once installed, type "jupyter-lab" in the terminal to open Jupyter Lab.

The first option is recommended for its ease to use, but any option is fine.

How to Use Jupyter Notebook Example

- Open the Jupyter notebook file in the Jupyter notebook program.
- Click "run" in Jupyter notebook.
- The Jupyter notebook file first asks the user to input the "user_token" and "cell_id" at the very beginning. Type your "user_token" and "cell_id".
- The Jupyter notebook file should have enough instructions on how to proceed. Some notable things that the Jupyter notebook file describes include the difference between "Amplabs timeseries data" and "Amplabs cycle data," and the fact that the user only needs to define 3 things to get the data plots they want: data point number limit, x-axis column name, y-axis column name.

The main purpose of the Client is to provide easy access to the data for users. The Client includes tools with which a user can simply select what battery data they want to plot, and then the tools output that battery data plot.

Access the Amplabs Client

CHAPTER

FOUR

MILESTONES

Standard data tools and practices built from the data hub standards will create a new data app marketplace.