

---

---

# Lean-stack data engineering with datafusion and more

— QP Hou @ Neuralink —

---

---

# \$> whoami

Name: QP Hou ([about.houqp.me](https://about.houqp.me))

Background:

- Software engineering at Neuralink
- Full stack from kernel to CSS
- Open-source hacker ([github.com/houqp](https://github.com/houqp)):
  - [delta-rs](#) author
  - [roapi](#) author
  - [Apache Arrow and Datafusion](#) PMC
  - [Apache Airflow](#) committer
  - [KORreader](#) maintainer
  - ...

# Neuralink

## Implant

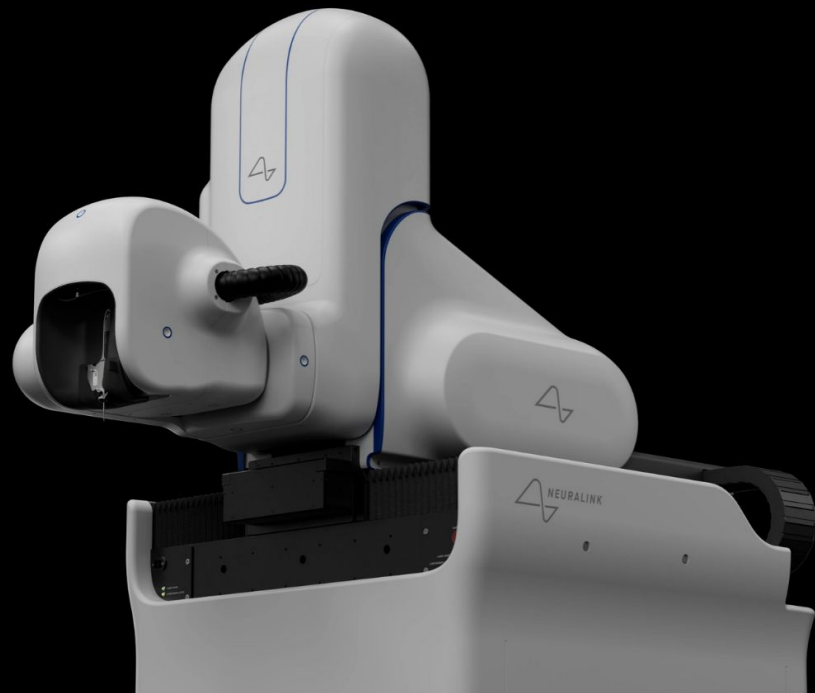
Our brain-computer interface is fully implantable, cosmetically invisible, and designed to let you control a computer or mobile device anywhere you go.



# Neuralink

## Surgical Robot

The threads of our implant are so fine that they can't be inserted by the human hand. Our surgical robot has been designed to reliably and efficiently insert these threads exactly where they need to be.





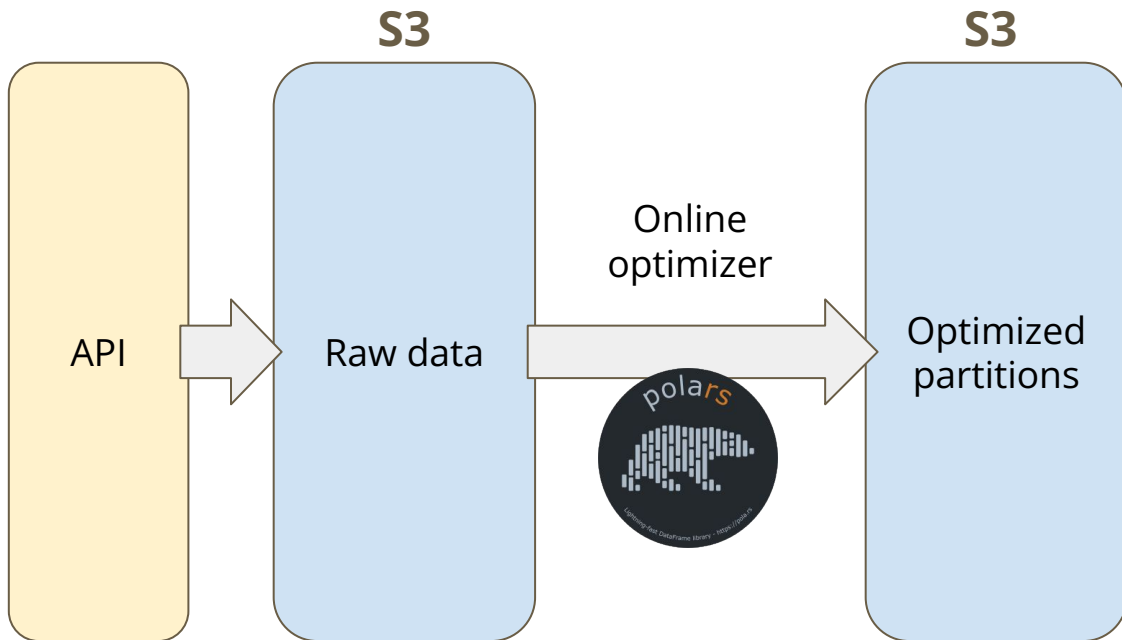
# Implant data pipeline



Neuron spikes



Implant telemetry

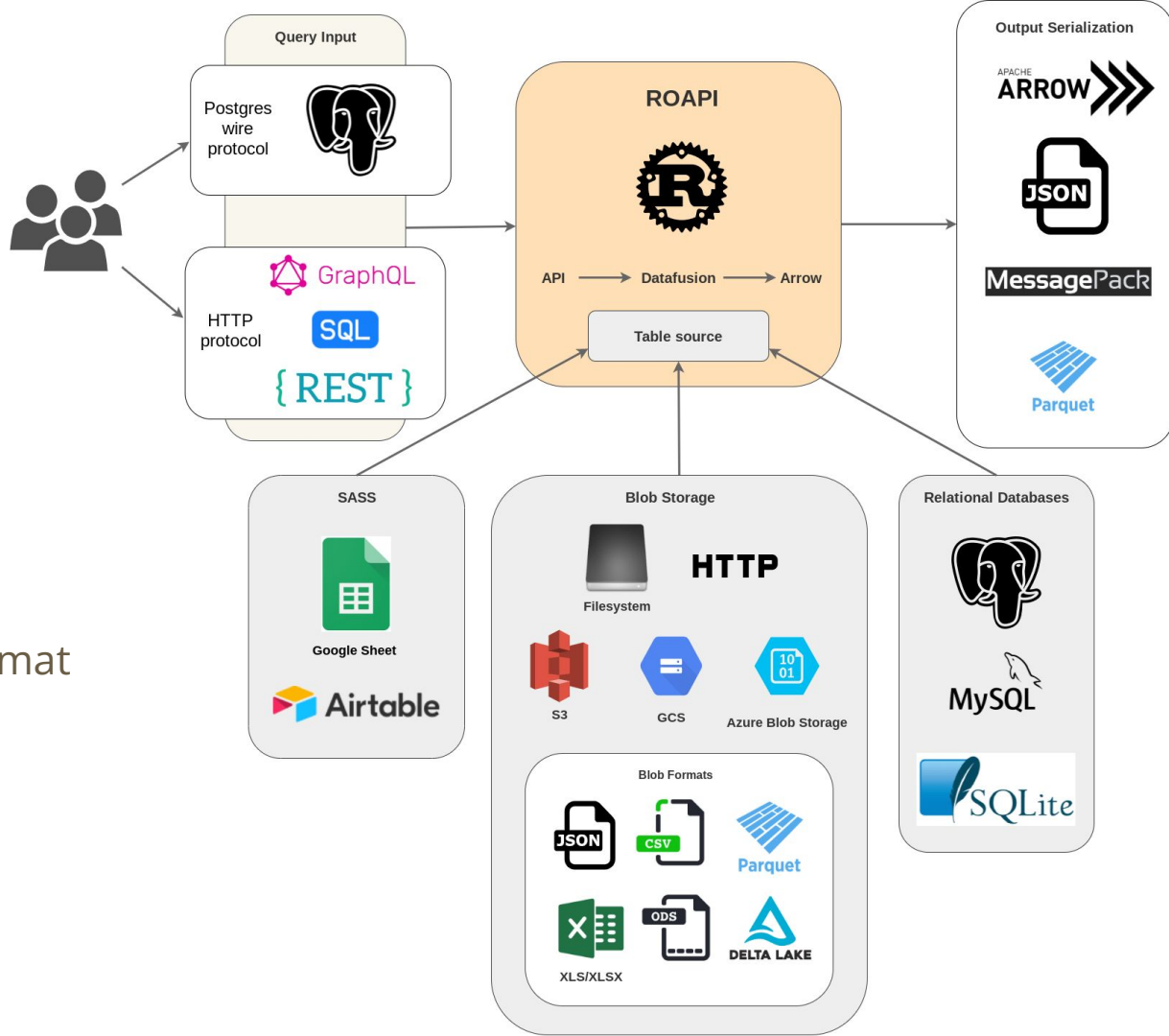


# ROAPI

[github.com/roapi/roapi](https://github.com/roapi/roapi)

A bridge for:

- Pluggable query interface
- Pluggable storage
- Pluggable serialization format



# ROAPI

```
1 addr:
2   http: 0.0.0.0:8084
3   postgres: 0.0.0.0:5432
4   flight_sql: 0.0.0.0:8888
5 tables:
6   - name: "telem"
7     uri: "s3://output-bucket/implant-events-parsed"
8     option:
9       format: "parquet"
10      use_memory_table: false
11      partition_columns:
12        - name: "implant_id"
```



# ROAPI

```
[2024-03-25T17:45:05Z INFO roapi::startup] 🚀 Listening on 0.0.0.0:5433 for Postgres traffic...  
[2024-03-25T17:45:05Z INFO roapi::startup] 🚀 Listening on 0.0.0.0:8888 for FlightSQL traffic...  
[2024-03-25T17:45:05Z INFO roapi::startup] 🚀 Listening on 0.0.0.0:8080 for HTTP traffic...
```

# ROAPI

```
curl -X POST -d "SELECT city, lat, lng FROM uk_cities LIMIT 2"  
localhost:8080/api/sql
```

```
curl -X POST -d "query { uk_cities(limit: 2) {city, lat, lng} }"  
localhost:8080/api/graphql
```

```
curl "localhost:8080/api/tables/uk_cities?columns=city,lat,lng&limit=2"
```

# Self-serviced dashboarding

Grafana FlightSQL datasource plugin ([github.com/influxdata/grafana-flightsql-datasource](https://github.com/influxdata/grafana-flightsql-datasource))



Grafana



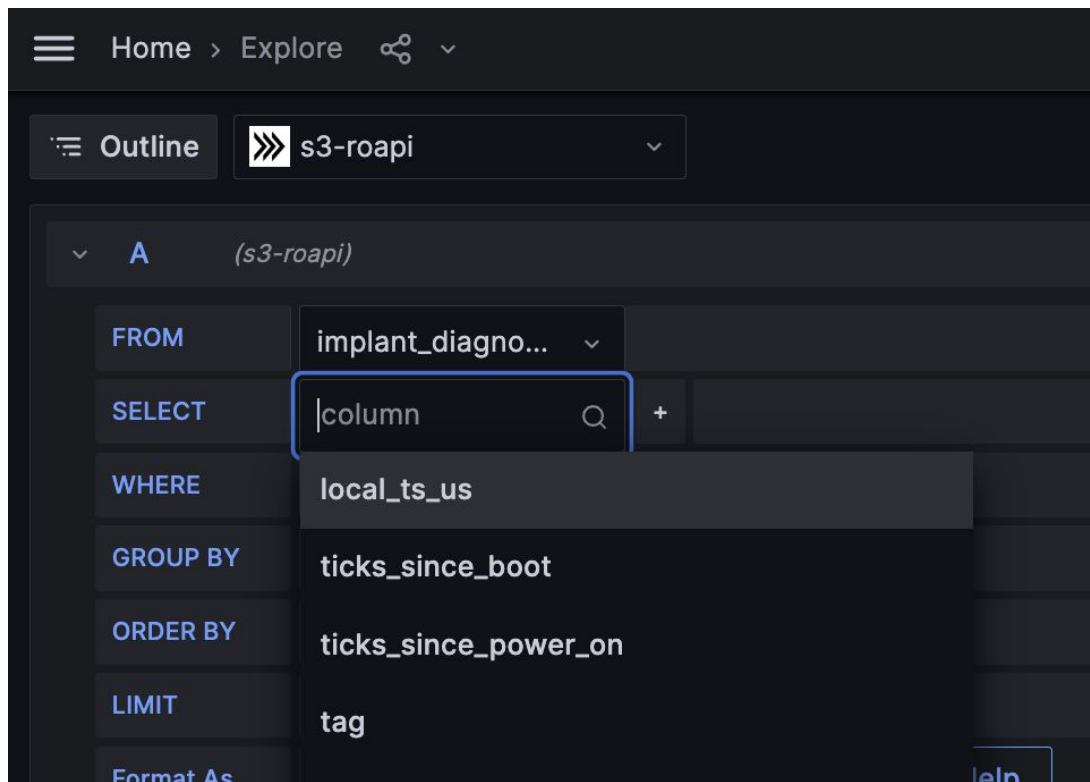
APACHE

**ARROW**

FLIGHT



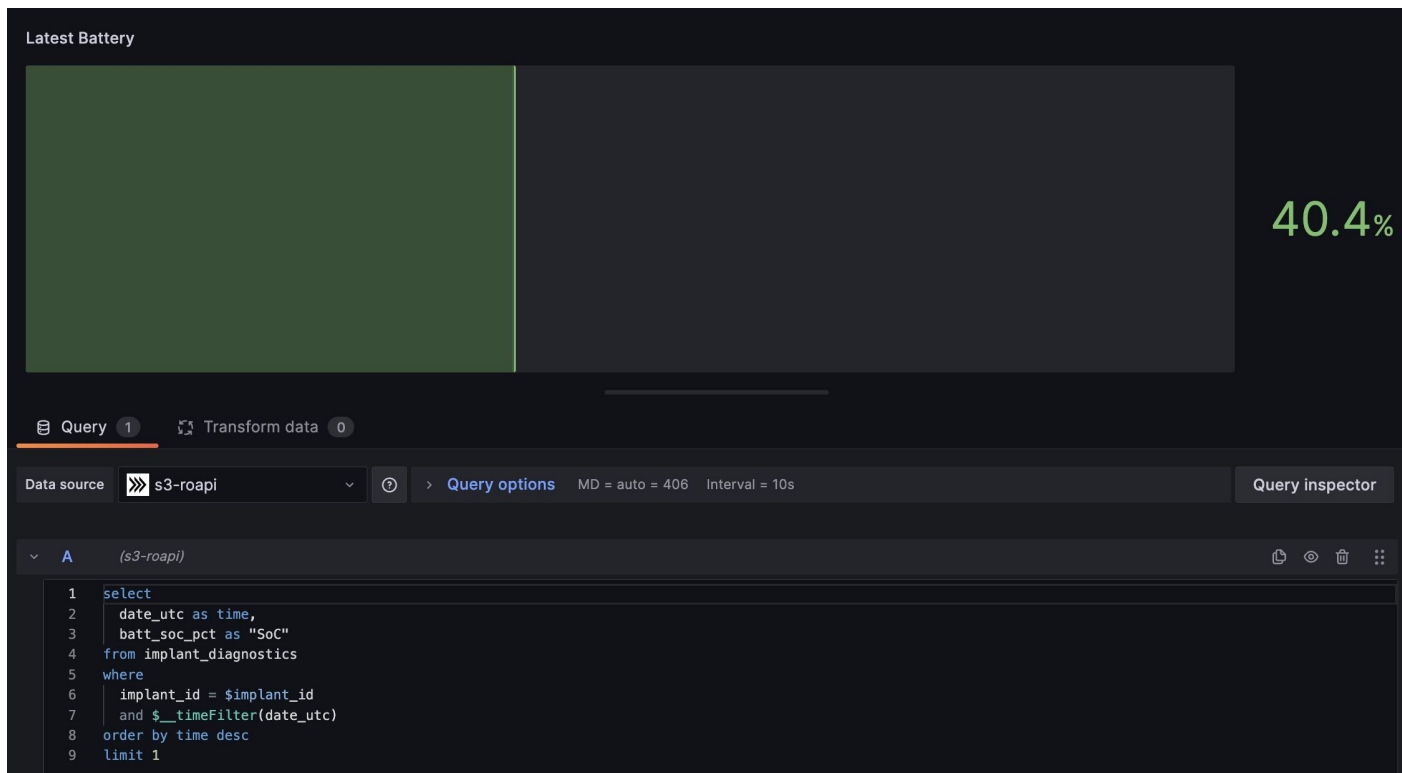
# Self-serviced dashboarding



The screenshot displays a self-serviced dashboarding interface. At the top, there is a navigation bar with a hamburger menu icon, the text "Home > Explore", and a share icon. Below this, there is a section with a "Outline" button and a dropdown menu showing "s3-roapi". The main area shows a query editor with a dropdown menu for column selection. The dropdown menu is open, showing a search bar with the text "column" and a magnifying glass icon. Below the search bar, there is a list of columns: "local\_ts\_us", "ticks\_since\_boot", "ticks\_since\_power\_on", and "tag". The "FROM" clause is set to "implant\_diagno...".

Clause	Value
FROM	implant_diagno...
SELECT	column
WHERE	local_ts_us
GROUP BY	ticks_since_boot
ORDER BY	ticks_since_power_on
LIMIT	tag

# Self-serviced dashboarding



**Latest Battery**

40.4%

**Latest Firmwa** **Latest Build Info**

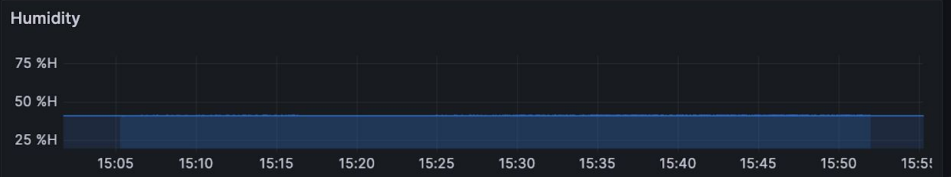
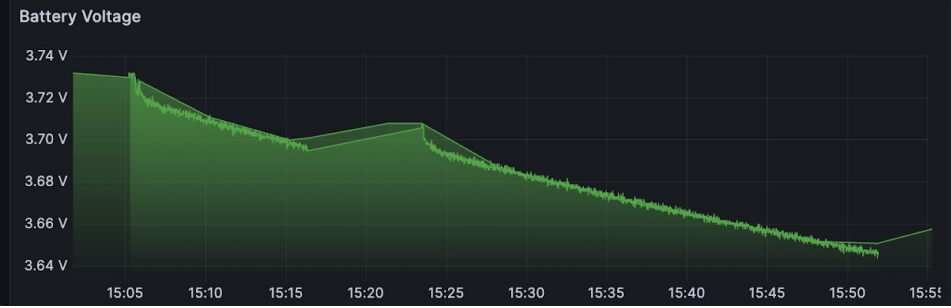
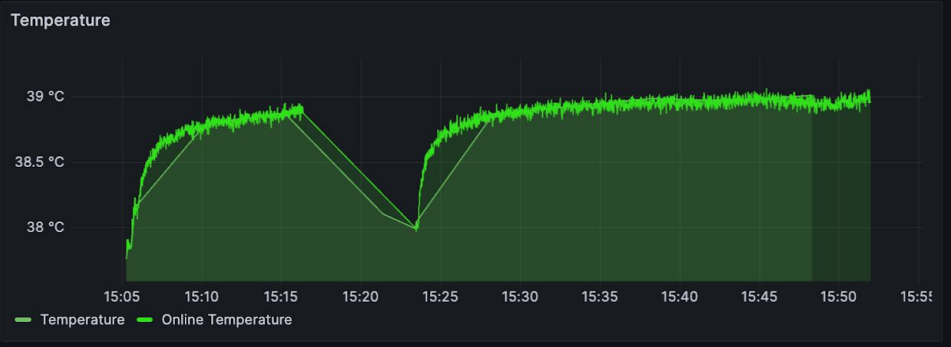
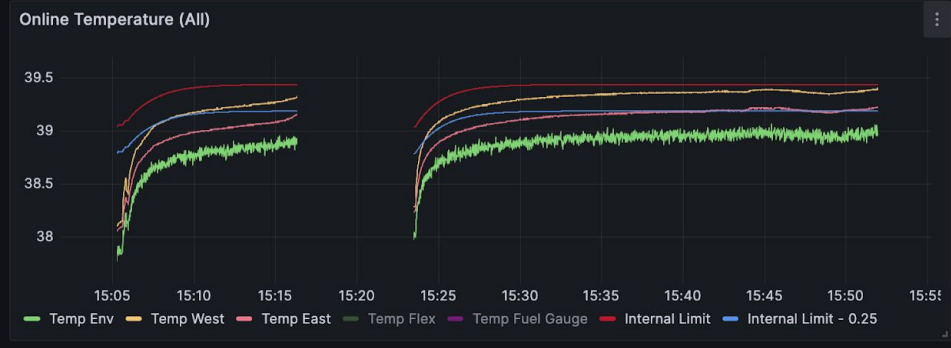
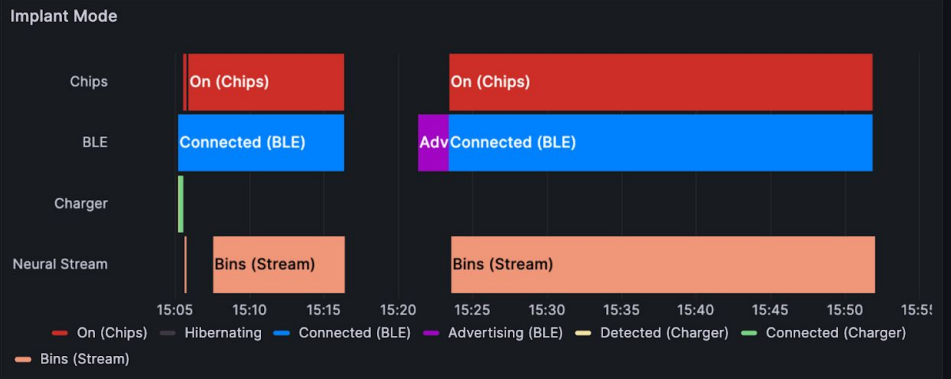
92 92+g93bac89

**Autowake peri** 1.75 hours

**Policy** RND

**Chargers** Charger ID 325

**Hibernation TI** 15 min **Hibernation TI** 15 min



# Optimize with datafusion table provider

Trait `datafusion::datasource::provider::TableProvider` 

[source](#) · [-]

```
pub trait TableProvider: Sync + Send {
    // Required methods
    fn as_any(&self) -> &dyn Any;

    fn schema(&self) -> SchemaRef;

    fn table_type(&self) -> TableType;

    fn scan<'life0, 'life1, 'life2, 'life3, 'async_trait>(
        &'life0 self,
        state: &'life1 SessionState,
        projection: Option<&'life2 Vec<usize>>,
        filters: &'life3 [Expr],
        limit: Option<usize>
    ) -> Pin<Box<dyn Future<Output = Result<Arc<dyn ExecutionPlan>>>> + Send + 'async_trait>>
    where Self: 'async_trait,
           'life0: 'async_trait,
           'life1: 'async_trait,
           'life2: 'async_trait,
           'life3: 'async_trait;
```

## Optimize with datafusion table provider

Query execution time reduced from **minutes** to **<100ms** with a custom provider that can leverage the internal index



# Data @ Neuralink

## Time series



Brain neuron spikes



Implant & Charger telemetry



## Multi-media

image/video/volumetric/etc.

# Data @ Neuralink

## Time series



Brain neuron spikes



Implant & Charger telemetry



Robot & Surgery



Manufacturing lines

## Multi-media

image/video/volumetric/etc.

# Data @ Neuralink

## Time series



Brain neuron spikes



Implant & Charger telemetry



Robot & Surgery

Manufacturing lines



## Multi-media

image/video/volumetric/etc.

Histopathology

Medical imaging

# Future work

- Multi-media data type support
- More query interfaces
  - Starlark, LLM, better drag and drop UI, etc.
- Storage layer query push-down
  - Custom S3 API extension powered by datafusion
- Custom table providers
- UDFs in WASM
- Data catalog platform

# What makes a better datafusion DX

- Stable API
  - Aligning API versions across dependencies can be time consuming
    - [delta-rs](#)
    - [convergence](#)
    - [connectorx](#)
- Documentation
  - I almost always end up reading the source code

# EOF

[neuralink.com/careers](https://neuralink.com/careers)