SithasoGunDB

Peer to Peer Database
with NodeJS and GunDB.

Table of contents

Welcome

Welcome to Peer to Peer Database with NodeJS and GunDB.

Here we will learn how to create a Peer to Peer Database using NodeJS and GunDB. We need to install NodeJS on our computer and then set up a peer relay server.

What is NodeJS?

Node.js is a cross-platform, open-source JavaScript runtime environment that can run on Windows, Linux, Unix, macOS, and more. Node.js runs on the V8 JavaScript engine, and executes JavaScript code outside a web browser.

While our database could hypothetically run without any kind of additional servers, this is going to require a sufficient number of users to make sure that you can always access enough nodes in the network.

To account for this, we're going to host our own relay peer that users can connect to even if no one else is using the app.

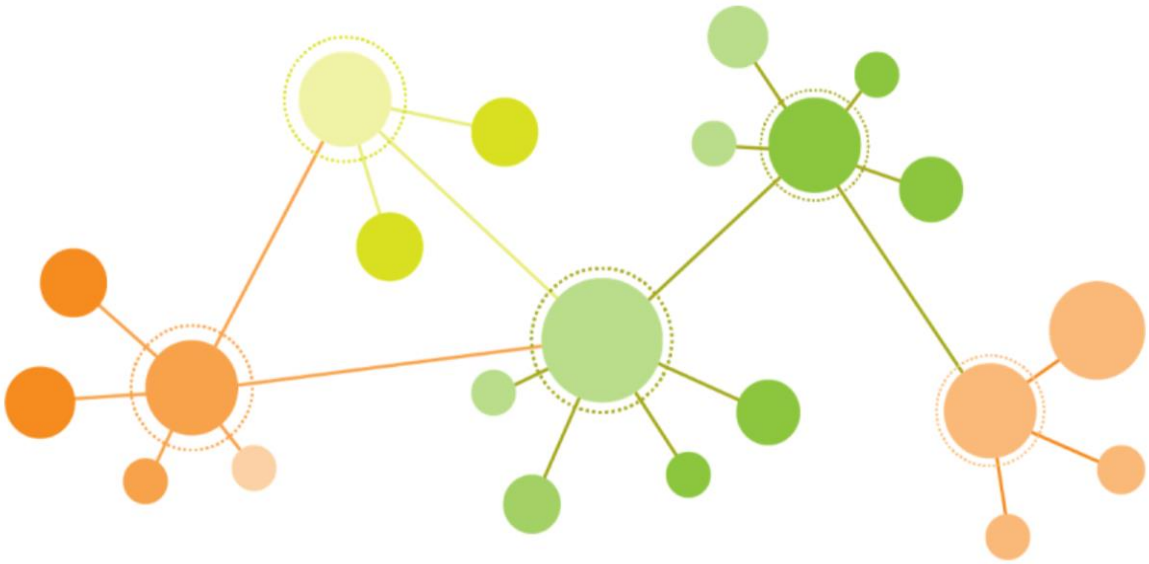
In terms of data formatting, every node in our database is going to have a 'soul' which is its unique identifier, and then data stored in a standard JSON format.

What is GunDB?

GunDB is an easy-to-use peer-to-peer decentralized database that will allow you to store data on a network of individual users, instead of a singular server. Each peer in the app's network stores some amount of the web app's data, but in aggregate, the entire network will contain all the necessary information.

The data can be stored in localStorage or IndexedDb on the users device and will be synced on connection to other peers. One is able able to subscribe to changes to data stored on the peers.

While this approach to decentralized data storage has many similarities to Blockchain, it is important to note that it is its own technology. In fact, just like Blockchain, decentralized databases are a hot topic in Computer Science research. You can learn more about GunDB here: https://gun.eco/



Create Peer Relay Server with NodeJS

To create a Peer Relay Server with NodeJS

1. Install NodeJS. You can download it here
2. Create a folder on your computer / new npm project
3. Open PowerShell and CD to this directory. In our example, this folder is called gun-peer-relay.
4. In this folder enter 'npm install express gun' and press enter to execute it. This installs express and gun on your folder.
5. Create a new js file in the folder, in our example, we have called it gun_server.js
6. This file has this content

```
const express = require('express')
const Gun = require('gun')

const app = express()
const port = 8000
app.use(Gun.serve)

const server = app.listen(port, () => {
    console.log("Listening at: http://localhost:" + port)
})

Gun({web: server})
```

7. Start the peer relay by entering 'node gun_server' and pressing enter.

Your peer relay will be running on http://localhost:8000/

You will find all this in the download.

Creating our App

We will create an App using BANano using the SithasoGunDB b4xlib.

In your code

1. Define the SithasoGunDB instance

```
Public gun As SithasoGUNDB
```

2. Then Initialize, AddPeers and Connect to it. In our example, we do this in our BANano_Ready call.

In the example below, we have a "people" parent node. We then add a record there with a "soul" i.e. unique id of "anele". This item has some properties like dob, firstname, lastname etc.

```
' HERE STARTS YOUR APP
Sub BANano_Ready()
'initialize gun db
gun.Initialize(Me, "gun")
'add peer to connect to
gun.AddPeer("http:localhost:8000/gun")
'connect to peers
gun.Connect
```

CREATE

This is done by the **put** call.

```
'CREATE
Dim rec As Map = CreateMap()
rec.Put("dob", "1973-15-04")
rec.Put("firstname", "Anele")
rec.Put("lastname", "Mbanga")
Dim bp As Object = BANano.Await(gun.PUT("people", "anele", rec))
Log(bp)
```

another example would be..

```
BANano.Await(gun.PUT("people", "usibabale", "Usibabale Mbanga"))
BANano.Await(gun.PUT("people", "olothando", "Olothando Mbanga"))
BANano.Await(gun.PUT("people", "esona", "Esona Mbanga"))
```

READ

This is done by the **get** call

```
'READ
Dim resY As Object = BANano.Await(gun.GET("people", "anele"))
Log(resY)
```

UPDATE

This is also done by the **put** call. We update a property and execute put again

```
'UPDATE
Dim rec1 As Map = CreateMap()
rec1.Put("town", "East London")
Dim bp As Object = BANano.Await(gun.PUT("people", "anele", rec1))
Log(bp)
```

DELETE

Our deletes are soft-deletes, on the record, we add a property named **'delete'** and we set it to true

```
'DELETE
Dim bp As Object = BANano.Await(gun.REMOVE("people", "anele"))
Log(bp)
```

To exclude deleted records, just detect if delete = true and if not include it.

GET_ALL

To extract all records saved in the database, one executes GET_ALL.

```
'GET_ALL
Dim people As List = BANano.Await(gun.GET_ALL("people"))
Log(people)
```

This gives us a list of available people

SUBSCRIBING

To be able to trap changes to records, one can subscribe to them. This will list available records and also list them anytime changes are made to the record.

```
'subscribe to a collection
gun.SUBSCRIBE_TABLE("people")
```

This is trapped with the following callback, eventName(gun)_subscribe(keyword)_people(tableName)

```
Private Sub gun_SUBSCRIBE_people (data As Map, id As String)
	Log("gun_SUBSCRIBE_people...")
	Log(data)
	Log(id)
End Sub
```