# **FROM 31st July TO 6th August**

# **Project ID:**

# **2021J\_BV01\_BCI Browser**

# **Project Title:**

# **Design and development of Brain Computer Interface Browser on Web and Mobile**

# **Summary:**

P300 spellers are among the most popular types of brain–computer interfaces (BCIs) and are extremely useful assistive devices that enable severely disabled patients to communicate. However, P300 speller performances should be further improved to translate laboratory designs into practical applications. We aimed to design a new speller paradigm that could evoke higher event-related potentials (ERPs) than traditional P300 spellers, thus improving the performance of BCI systems.

# **Detail:**

**POSTGRESQL**

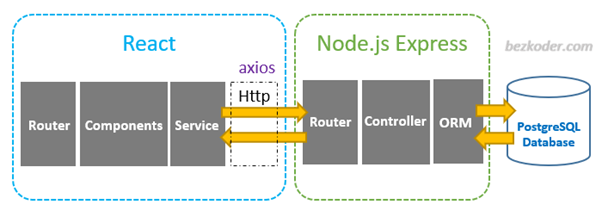
Postgresql is a popular SQL database. It has been in active development for the last 30+ years and is considered to be one of the most advanced relational databases out there. PostgreSQL is also easy to learn and setup compared to other relational databases available. Because of its free and open-source nature, this is a popular choice among startups.

PostgreSQL is a cross-platform database that runs on all major operating systems. The configuration and access/database creation slightly differs between operating systems, though.

Postgresql is a widely used system for Object-Relational Database Management that is mainly used for large web applications. It provides additional and substantial power by incorporating four basic concepts in such a way that the user can extend the system without any problem. It extends and uses the SQL language that is combined with various features to safely scale and store the intricate data workloads.

Following are some of the major features of Postgresql :

1. Object-relational database
2. Supports major Operating systems
3. Support Extensibility for SQL and Complex SQL queries
4. Nested transactions
5. Flexible API and Database validation
6. Multi-version concurrency control (MVCC) and Procedural languages
7. WAL and Client server
8. Table inheritance & Asynchronous replication



**Project Configuration**

Let's get started with a simple blank Node.js project with the default settings:

$ npm init -y

Then, let's use npm to install the node-postgres module, which will be used to connect to and interact with Postgres:

$ npm install --save pg

This is an example index.js file that I put into a project I generated with npm init and installed node-postgres into:

const { Pool, Client } = require("pg");

const credentials = {

user: "postgres",

host: "localhost",

database: "nodedemo",

password: "yourpassword",

port: 5432,

};

// Connect with a connection pool.

async function poolDemo() {

const pool = new Pool(credentials);

const now = await pool.query("SELECT NOW()");

await pool.end();

return now;

}

// Connect with a client.

async function clientDemo() {

const client = new Client(credentials);

await client.connect();

const now = await client.query("SELECT NOW()");

await client.end();

return now;

}

// Use a self-calling function so we can use async / await.

(async () => {

const poolResult = await poolDemo();

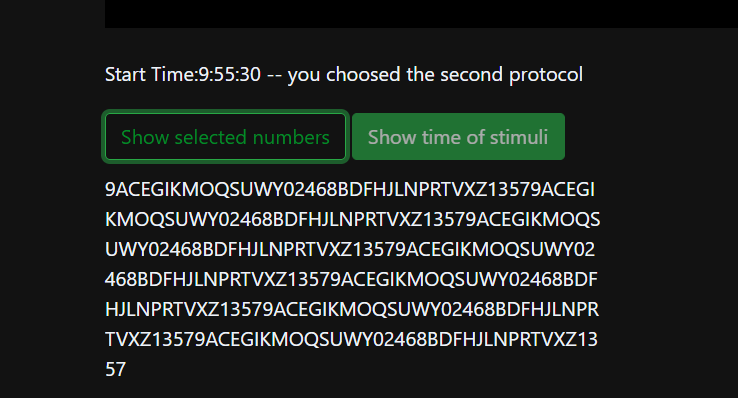
console.log("Time with pool: " + poolResult.rows[0]["now"]);

const clientResult = await clientDemo();

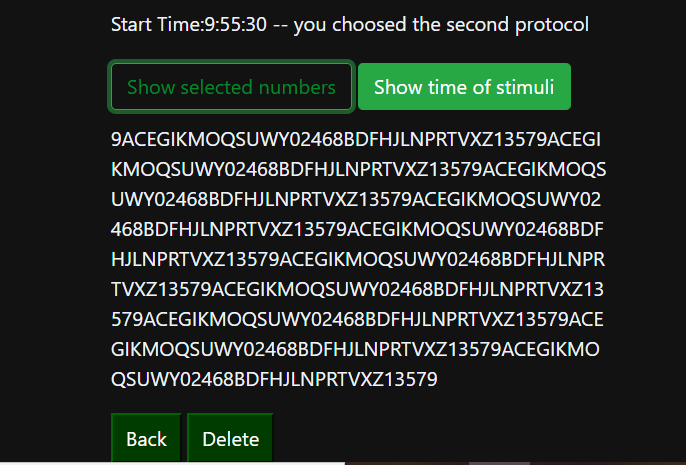
console.log("Time with client: " + clientResult.rows[0]["now"]);

})();

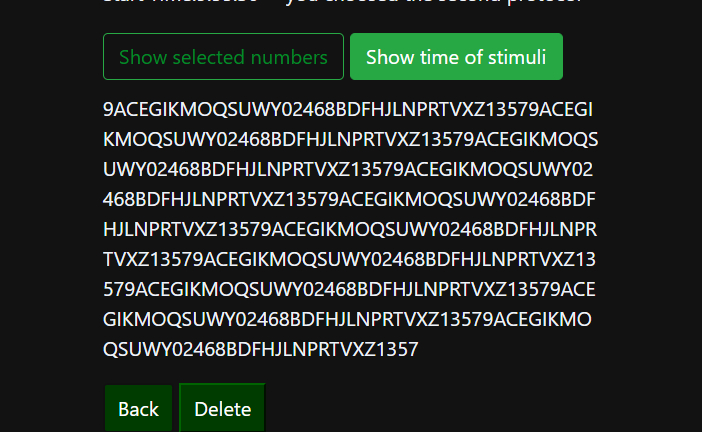
**UPDATES ON P300 SPELLER PROJECT**



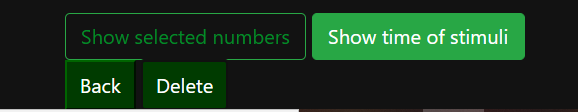
The number/letter that is flashing is displayed if you click Show Selected Numbers.



Delete and Back buttons have been added.



As you can see, 9 has been removed using the back button



All numbers have been deleted using the delete button.