✨ Member-only story

ANDROID DATABASE CONNECTION

How To Connect An Android Project To A Postgresql Database

Connecting android projects to a PostgreSQL database by using JDBC

Mustafa Katipoğlu · Follow

Published in Cyber Explorer
4 min read · Nov 30, 2020

Listen Share

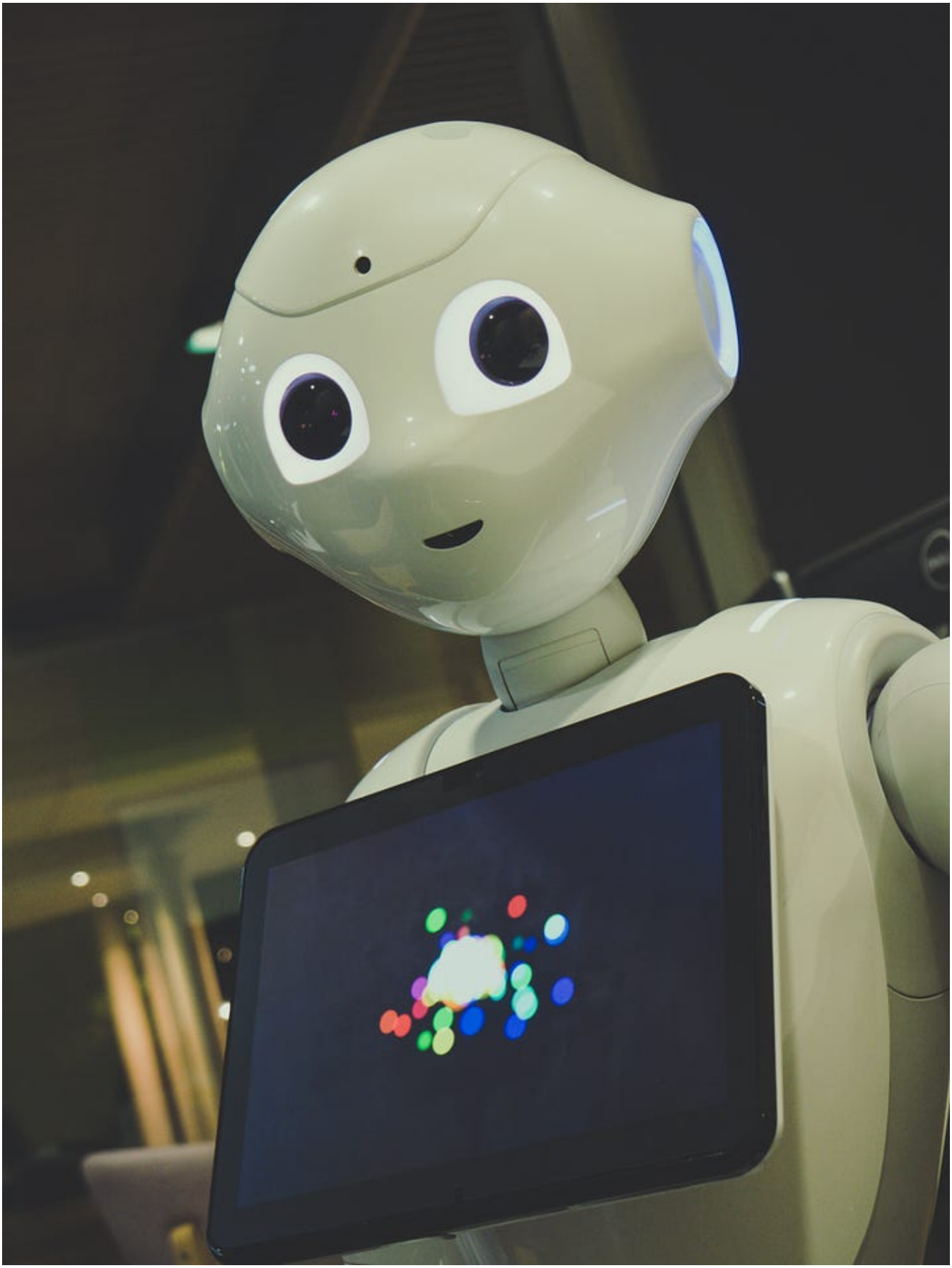


Photo by Owen Beard on Unsplash

Do you want to connect your database to your android application? Here in this article, I would like to show you how to connect a PostgreSQL database to an

android studio project. I also show the integration of Local, Amazon Cloud or Google Cloud PostgreSQL service to your android application.

Ways to connect Android project to a database

There are two ways of connecting an Android project to a PostgreSQL project.

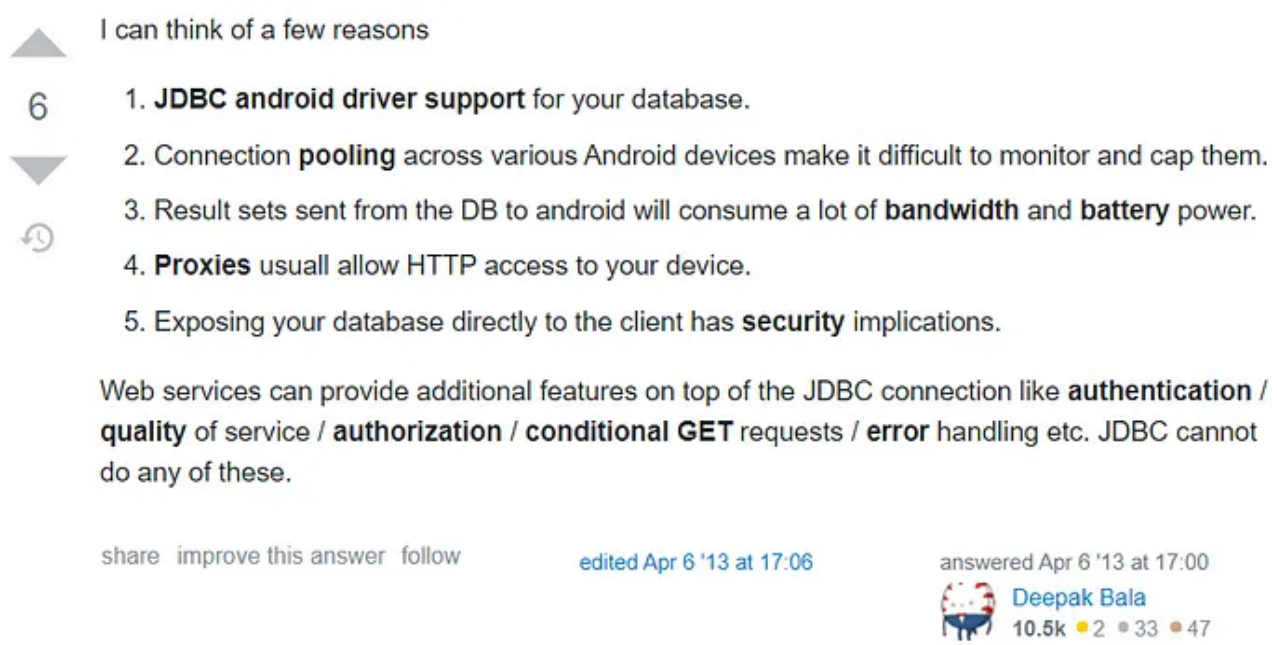
1. Using Web Services
2. Using JDBC

When we search the web how people normally connect their android apps to a database, we find that most of them serve the database requests over a web service API.

Why serve database requests over API rather than using JDBC:

- Increased speed
- High security
- Lower power consumption
- Easy monitoring of database requests

While serving database requests by using a web service is a perfect solution and has lots of advantages over using JDBC, it is a bit more complicated to set up at first.



Why use Web Services over JDBC For Android — Screenshot by Author

How to connect an Android project to PostgreSQL using JDBC driver

Using Web Services is a better choice when it comes to connecting an Android to a database. But, for those who need to prototype their database in the development phase, creating API may seem unnecessary. That was the case for me.

Here is how I connected the PostgreSQL database to my android project using JDBC driver.

Add PostgreSQL JDBC Driver Dependency

First, to make the PostgreSQL JDBC connection work, we need to set up the PostgreSQL JDBC driver by hand or better use Gradle dependency for the PostgreSQL JDBC driver.

Add the following dependency to your build.gradle(Module) file.

```
implementation 'org.postgresql:postgresql:42.2.5'
```

Add Network Permissions

In order to connect to the Postgresql database instance, the android application needs to have permission to access the Internet and Network State.

Add the following two uses-permission statements to your project.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.teknoarktik.android.greenhouse">

    <!-- Put the following two uses permissions statemens for remote db connection -->

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.GreenHouse">
        <activity android:name=".MainActivity">
            <intent-filter>
```

```
19                <action android:name="android.intent.action.MAIN" />
20
21                <category android:name="android.intent.category.LAUNCHER" />
22            </intent-filter>
23        </activity>
24    </application>
25 </manifest>
```

AndroidManifest.xml hosted with ♡ by GitHub view raw

Configure The JDBC PostgreSQL Connection

Configuring the JDBC connection is quite easy except for the host part of the configuration as it changes PostgreSQL vendor to vendor.

By default, PostgreSQL uses port 5432, the database name of 'postgres' with a user name 'postgres'. You can change any of the default configurations from the first setup screen of PostgreSQL or by using other means.

Amazon Postgresql Connection

The host variable has two options as you can see. If you are using the **Amazon Postgresql** service, you will probably have a host named like a subdomain of amazonaws website.

```
1  private final String host = "ssprojectinstance.csv2nbvvgbcb.us-east-2.rds.amazonaws.com"
2  private final String database = "postgres";
3  private final int port = 5432;
4  private final String user = "postgres";
5  private final String pass = "123456";
```

configure_android_amazon_cloud_postgresql_connection.java hosted with ♡ by GitHub view raw

Google Cloud Postgresql Connection

If you are using the **Google Cloud Postgresql** service, you will be given an IP as a host.

```
1  private final String host = "35.44.16.169";
2  private final String database = "postgres";
3  private final int port = 5432;
4  private final String user = "postgres";
5  private final String pass = "123456";
```

configure_android_google_cloud_postgresql_connection.java hosted with ♡ by GitHub view raw

Configure Android to Google Cloud PostgreSQL Connection

Localhost Postgresql Connection

First and foremost, to connect to your localhost PostgreSQL, you need to leave the PostgreSQL open.

The critical thing to remember when connecting to your localhost is that the simulator device has its own localhost. Therefore you can not use the local host IP(127.0.0.1).

Network Address	Description
10.0.2.1	Router/gateway address
10.0.2.2	Special alias to your host loopback interface (i.e., 127.0.0.1 on your development machine)
10.0.2.3	First DNS server
10.0.2.4 / 10.0.2.5 / 10.0.2.6	Optional second, third and fourth DNS server (if any)
10.0.2.15	The emulated device network/ethernet interface
127.0.0.1	The emulated device loopback interface

Android Emulator Networking By Android Dev Docs

When you are connecting your localhost to the emulator's localhost, you need to use the special IP provided by the android emulator.

```
1   private final String host = "10.0.2.2";
2   private final String database = "postgres";
3   private final int port = 5432;
4   private final String user = "postgres";
5   private final String pass = "123456";
```

configure_android_localhost_postgresql_connection.java hosted with ♡ by GitHub view raw

Configure Android to Local PC PostgreSQL Connection

Java Part of PostgreSQL Android Connection

Here we first create a new object called Database.

```
1   import java.sql.Connection;
2   import java.sql.DriverManager;
3
4   public class Database {
5
```

```
    private Connection connection;

    // For Amazon Postgresql
    // private final String host = "ssprojectinstance.csv2nbvvgbcb.us-east-2.rds.amazonaws.com"

    // For Google Cloud Postgresql
    // private final String host = "35.44.16.169";

    // For Local PostgreSQL
    private final String host = "10.0.2.2";

    private final String database = "postgres";
    private final int port = 5432;
    private final String user = "postgres";
    private final String pass = "123456";
    private String url = "jdbc:postgresql://%s:%d/%s";
    private boolean status;

    public Database()
    {
        this.url = String.format(this.url, this.host, this.port, this.database);
        connect();
        //this.disconnect();
        System.out.println("connection status:" + status);
    }

    private void connect()
    {
        Thread thread = new Thread(new Runnable() {
            @Override
            public void run()
            {
                try
                {
                    Class.forName("org.postgresql.Driver");
                    connection = DriverManager.getConnection(url, user, pass);
                    status = true;
                    System.out.println("connected:" + status);
                }
                catch (Exception e)
                {
                    status = false;
                    System.out.print(e.getMessage());
                    e.printStackTrace();
                }
            }
        });
```

```
        thread.start();
        try
        {
            thread.join();
        }
        catch (Exception e)
        {
            e.printStackTrace();
            this.status = false;
        }
    }

    public Connection getExtraConnection()
    {
        Connection c = null;
        try
        {
            Class.forName("org.postgresql.Driver");
            c = DriverManager.getConnection(url, user, pass);
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }

        return c;
    }
}
```

At the MainActivity, just create an instance of Database. When it calls the constructor, it will try to connect to the remote PostgreSQL database.

```
package com.teknoarktik.android.greenhouse;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Database db = new Database();
```

```
14        }
15    }
```

MainActivity.java hosted with ♡ by GitHub view raw

4- Run the application

When you run the application, take a look at the logcat.

If you see the following output at your logcat, then you have successfully connected to your remote PostgreSQL instance.

```
connected:true
```

But, probably you will see “The connection attemp failed error which is caused by Operation not permitted error”.

```
1   org.postgresql.util.PSQLException: The connection attempt failed.
2   at org.postgresql.core.v3.ConnectionFactoryImpl.openConnectionImpl(ConnectionFactoryImpl.java:2
3   at org.postgresql.core.ConnectionFactory.openConnection(ConnectionFactory.java:49)
4   at org.postgresql.jdbc.PgConnection.<init>(PgConnection.java:195)
5   at org.postgresql.Driver.makeConnection(Driver.java:454)
6   at org.postgresql.Driver.connect(Driver.java:256)
7   at java.sql.DriverManager.getConnection(DriverManager.java:580)
8   at java.sql.DriverManager.getConnection(DriverManager.java:218)
9   at com.teknoarktik.android.greenhouse.Database$1.run(Database.java:31)
10  at java.lang.Thread.run(Thread.java:919)
11  Caused by: java.net.SocketException: socket failed: EPERM (Operation not permitted)
12  at java.net.Socket.createImpl(Socket.java:492)
13  at java.net.Socket.connect(Socket.java:619)
14  at org.postgresql.core.PGStream.<init>(PGStream.java:70)
15  at org.postgresql.core.v3.ConnectionFactoryImpl.tryConnect(ConnectionFactoryImpl.java:91)
16  at org.postgresql.core.v3.ConnectionFactoryImpl.openConnectionImpl(ConnectionFactoryImpl.java:1
```

error hosted with ♡ by GitHub view raw

This exception is thrown because the application can not access the network state and internet. Even though we have given permission at the third step, it does fail to get the permission somehow.

In order to solve the connection attempt failed error, you need to delete the application from the emulator or your physical device and then reinstall it by

running the project again or however, you like to install it.

When you reinstall the application, it will be installed with the permissions you have specified in the AndroidManifest.xml file.

```
I/System.out: connected:true
I/System.out: connection status:true
```

The code is working now. — Screenshot by author

Android Development | Postgresql | Amazon Postgresql | Google Clouds Postgresql | Android Db Connection

Follow

Written by Mustafa Katipoğlu

154 Followers · Editor for Cyber Explorer

Computer Engineer

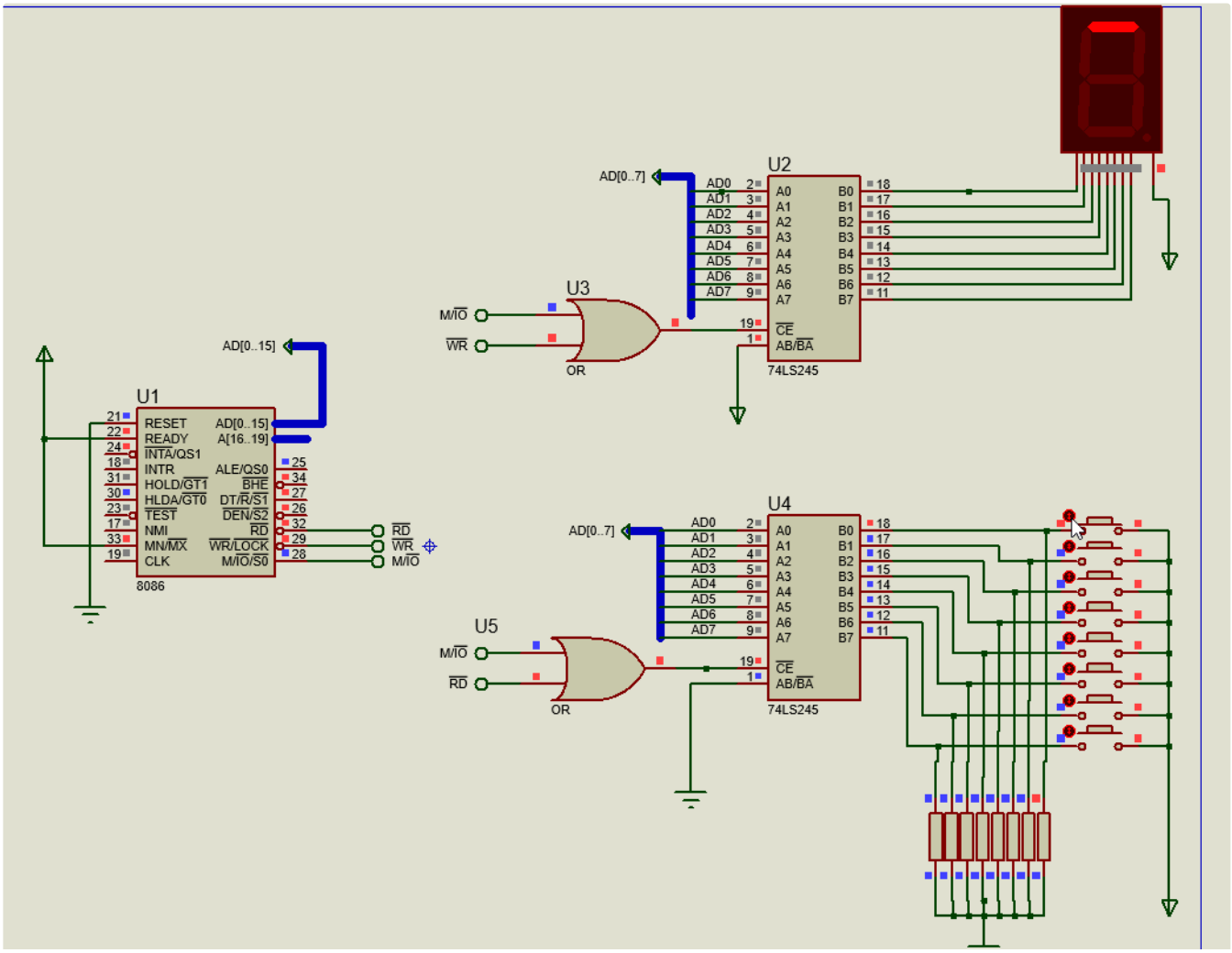
More from Mustafa Katipoğlu and Cyber Explorer

Mustafa Katipoğlu in Towards Data Science

Programming Path For Those Failing to Learn Programming

The path from the very beginning to the further high-quality programming practices.

10 min read · Nov 28, 2020



Learn 8086 Microprocessor With Proteus Circuits

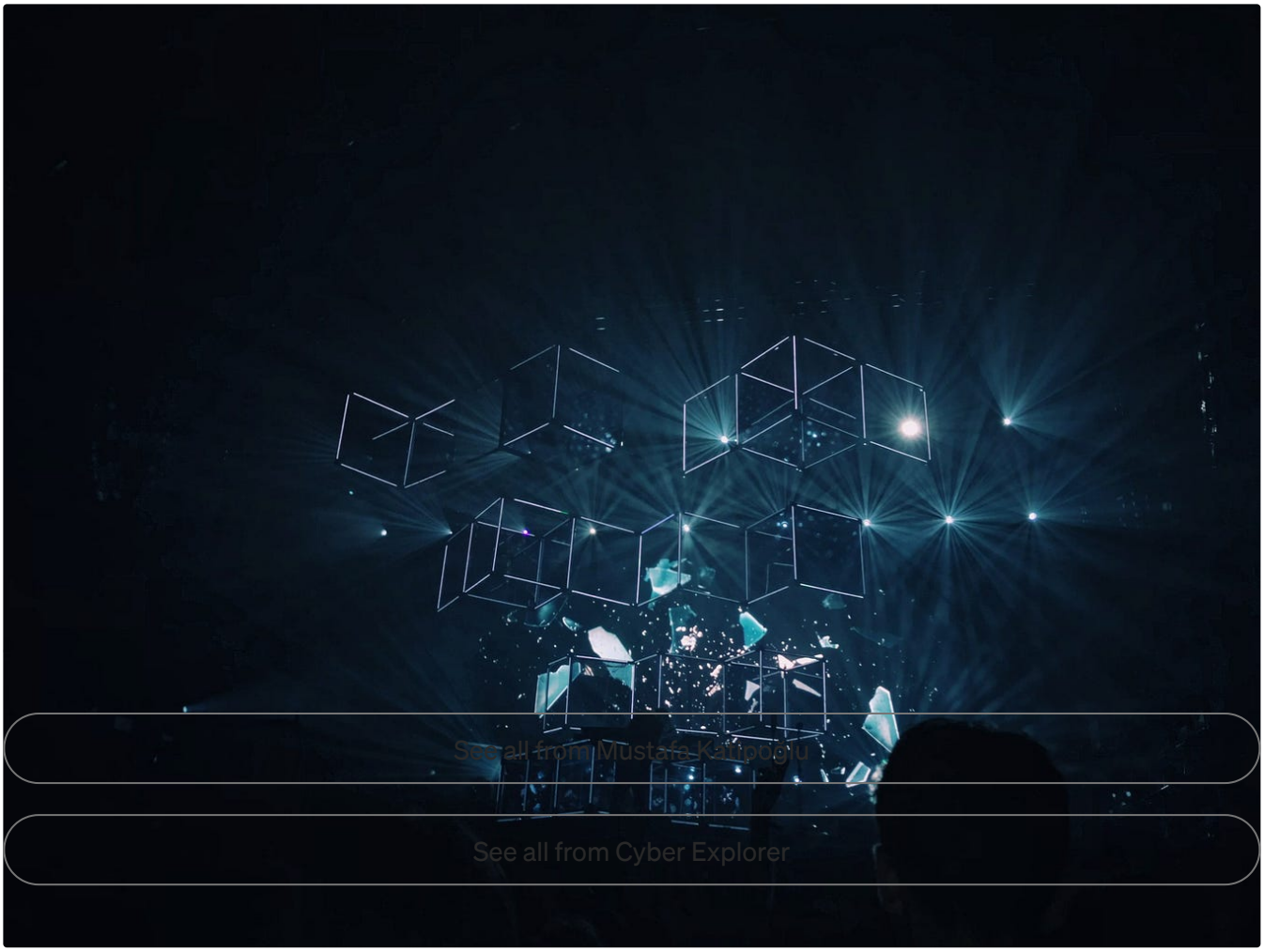
If you want to learn the 8086 microprocessors by designing simple circuits come closer. Here, I share my proteus 8086 designs for...

Mustafa Katipoğlu in Cyber Explorer

Java CEF Based Web Browser Development Environment Setup

Integrate the latest Java Chromium Embedded Framework(JCEF) into your web browser development environment.

· 3 min read · Sep 5, 2021



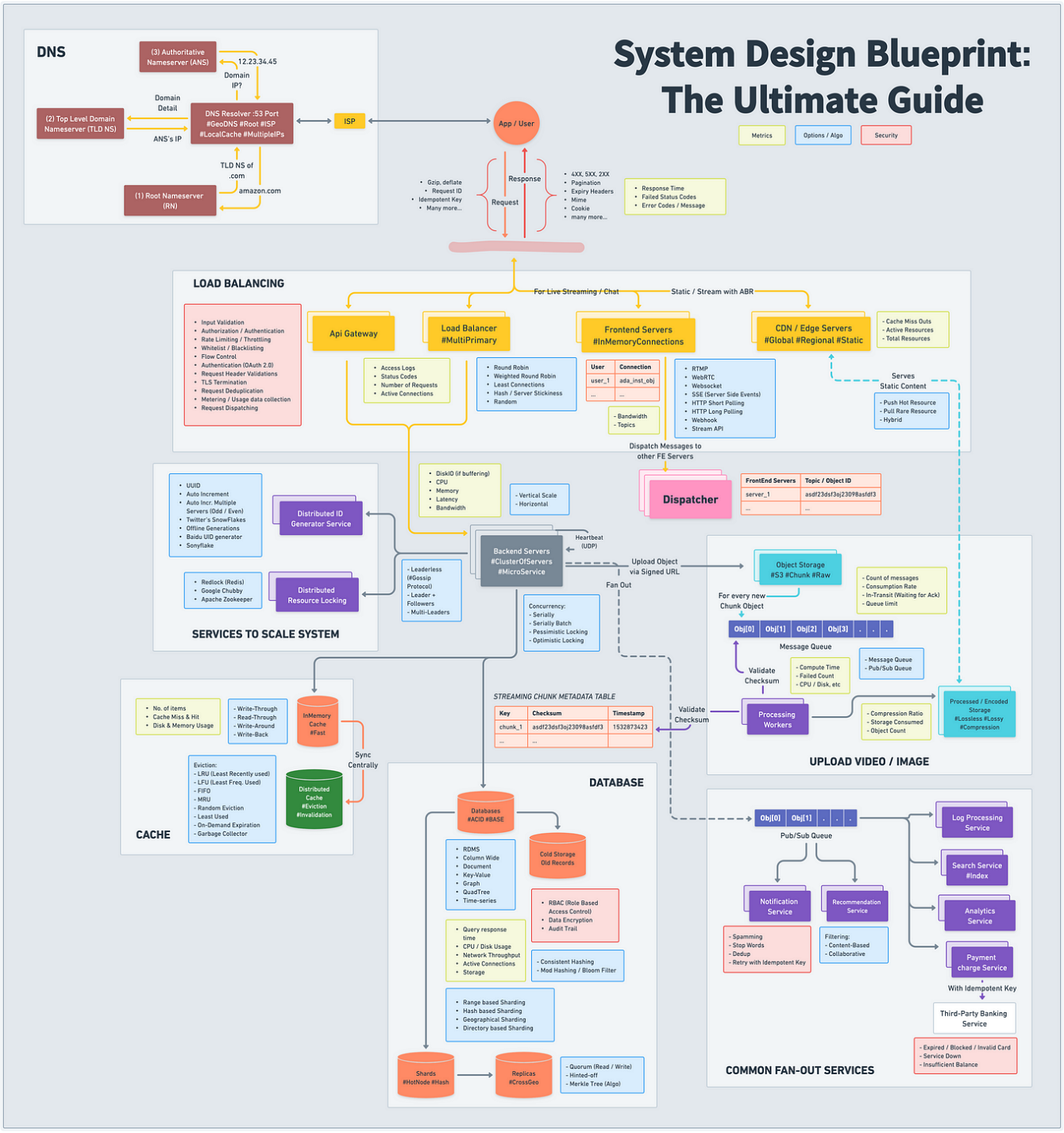
Mustafa Katipoğlu in Recommendation Systems

User-based vs Item-based Collaborative Filtering

Even though both user-based and item-based collaborative filtering algorithms are complementary and hybrid systems performs better, for...

✨ · 4 min read · Feb 17, 2020

30 1

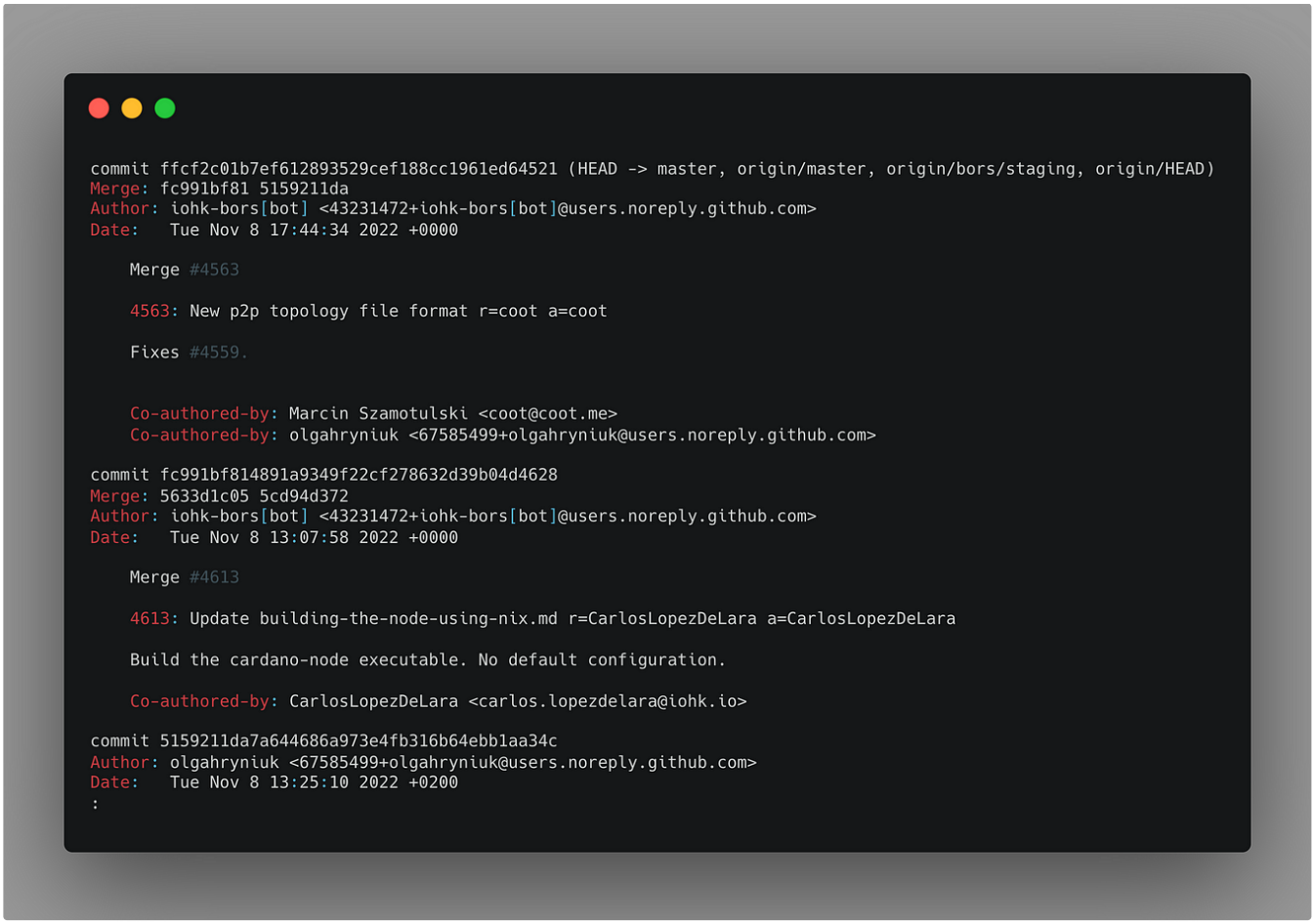


Love Sharma in Dev Genius

System Design Blueprint: The Ultimate Guide

Developing a robust, scalable, and efficient system can be daunting. However, understanding the key concepts and components can make the...

· 9 min read · Apr 20



Jacob Bennett in Level Up Coding

Use Git like a senior engineer

Git is a powerful tool that feels great to use when you know how to use it.

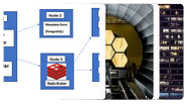
4 min read · Nov 15, 2022

6.7K 67

Lists

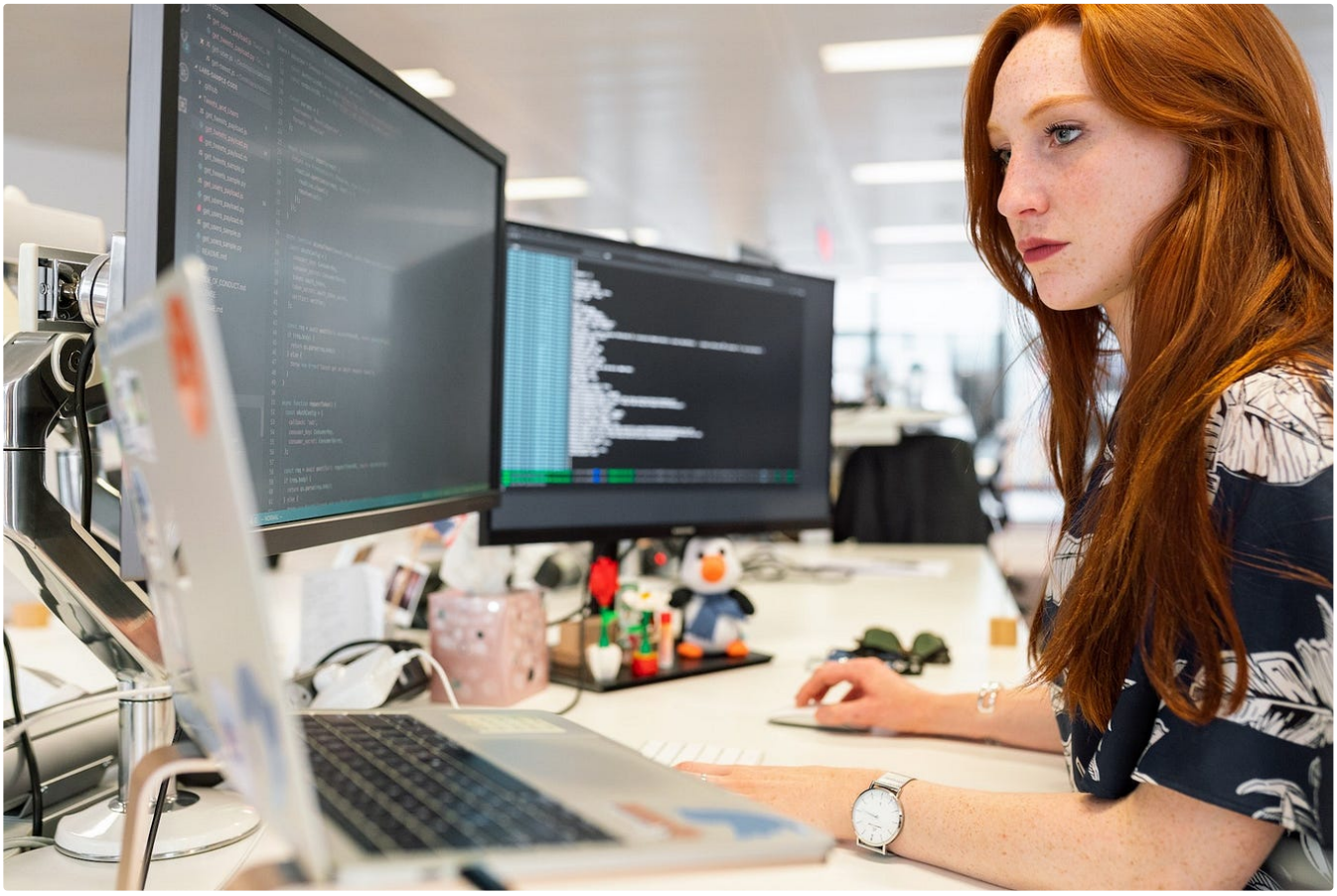
Now in AI: Handpicked by Better Programming

248 stories · 15 saves



New_Reading_List

173 stories · 7 saves



The Coding Diaries in The Coding Diaries

Why Experienced Programmers Fail Coding Interviews

A friend of mine recently joined a FAANG company as an engineering manager, and found themselves in the position of recruiting for...

· 5 min read · Nov 2, 2022

4.5K 99

JP Brown

What Really Happens to a Human Body at Titanic Depths

A Millisecond-by-Millisecond Explanation

· 4 min read · Jun 22

26K 312

Julie Perilla Garcia in Level Up Coding

To Be A Great Software Developer— You Need a System

Here's how to build one.

· 7 min read · Jun 23

288 3