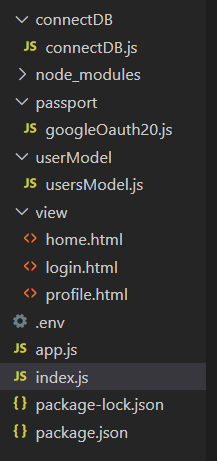
How to use googleOauth20 for google authentication

the necessary files are



**Creating Server and connect to mongoDB database:** create a index.js and app.js then install the necessary npm package like express, nodemon, dotenv, cors, mongoose

Before installing this package make sure u have initialize the npm package using npm init -y

After installing the necessary package create a server using app.listen(port, ()=>{})

After that, connect the express server to mongoDB database using mongoose

To connect the mongoDB database write the code in the connect.js

let mongoose=require("mongoose")

require("dotenv").config()

let db\_url=process.env.DB\_url

module.exports= mongoose.connect(db\_url)

.then(()=>{

    console.log("mongoDB is connected")

})

.catch(()=>{

    console.log("mongoDB is not connected")

})

Then create a users schema and users model for the mongoDB database in the usersModel.js

Here we create a username and googleId in the schema

const { MongoDriverError } = require("mongodb")

let mongoose=require("mongoose")

let usersSchema= mongoose.Schema({

    username:{

        type:String

    },

    googleId:{

        type:String

    }

})

module.exports= mongoose.model("usersInfo",usersSchema)

then import the connectDB.js in the index.js and write the code for creating the server

let app=require("./app")

require("dotenv").config()

let port=process.env.port  || 5000

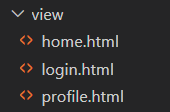
require("./connectDB/connectDB")

app.listen(port,()=>{

    console.log(`your server is running at http://localhost:${port}`)

})

**Creating views file:**



**Importing and using passport and googleOauth20 strategy:**

To use passport firstly, install the npm package named passport, express-session

And to store sessions we need to install connect-mongo package

And then require it to app.js and use it

let express=require("express")

Install all these package and then require it to use

let cors=require("cors")

let passport=require("passport")

let session=require("express-session")

let MongoStore=require("connect-mongo")

require("dotenv").config()

This is a required option

require("./passport/googleOauth20")

let app=express()

app.use(cors())

app.set('trust proxy', 1) // trust first proxy

app.use(session({

  secret: 'keyboard cat',

  resave: false,

Create a MongoStore to enter the mongoUrl and collectionName to store sessions

  saveUninitialized: true,

  store: MongoStore.create({

    mongoUrl:process.env.DB\_url,

    collectionName:"sessions"

  })

}))

Initialize both passport.initialize() and passport.session() in app

app.use(passport.initialize())

app.use(passport.session())

to know more about passport, express-session , mongoUrl

passport= <https://www.npmjs.com/package/passport>

express-session= <https://www.npmjs.com/package/express-session>

to store sessions we use connect-mongo store that is most popular use

connect-mongo = <https://www.npmjs.com/package/connect-mongo#install>

That is how we declare passport and sessions in the application and now we are ready to use strategies of passport. In this project, we use googleOauth20 for users authentication

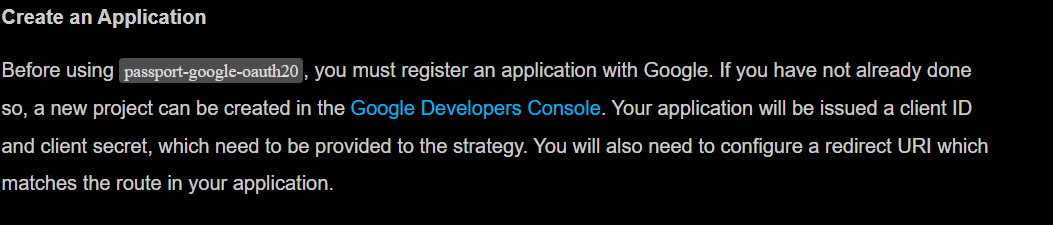
**Using googleOauth20 :**

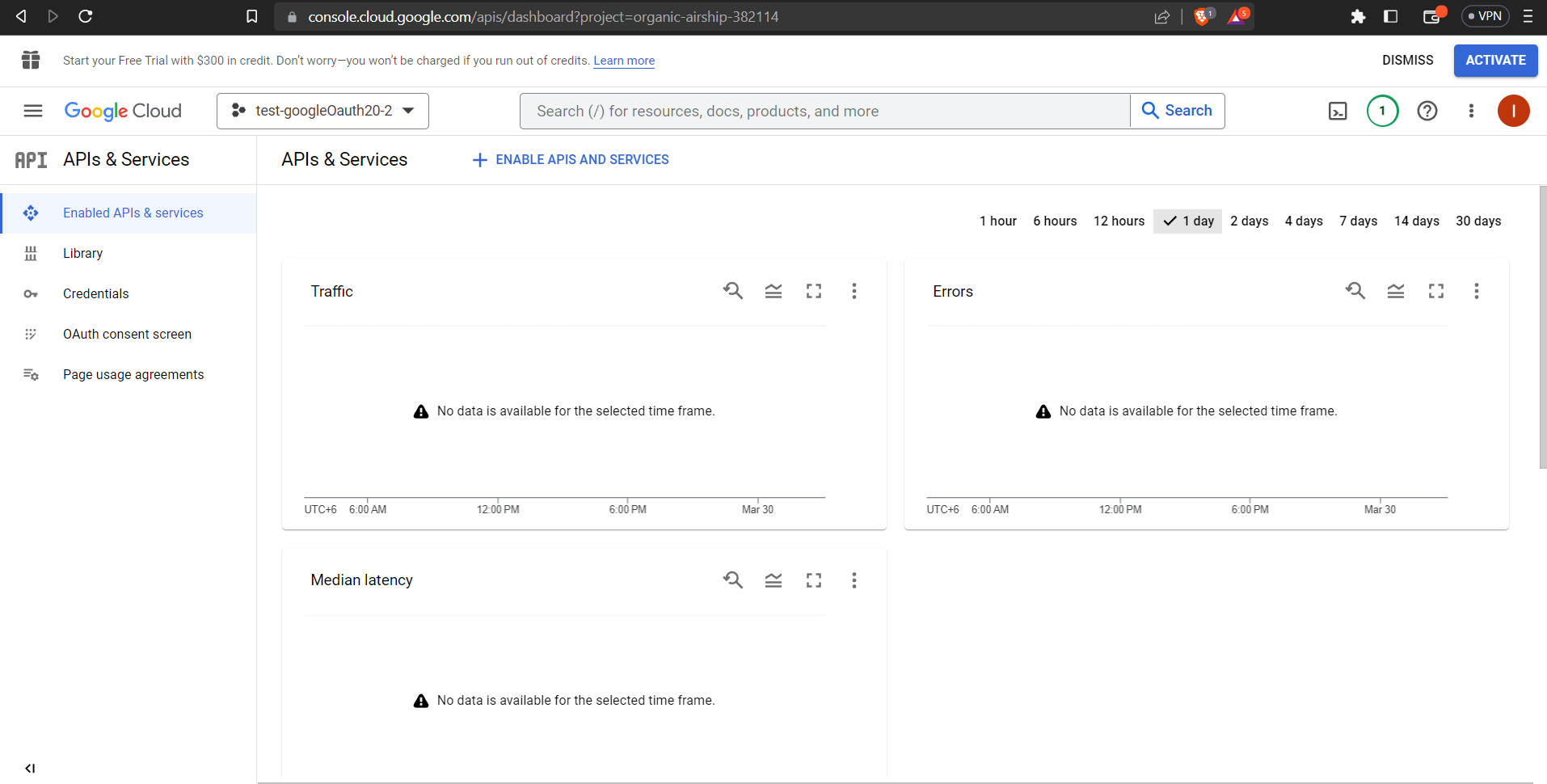
googleOauth20 = <https://www.passportjs.org/packages/passport-google-oauth20/>

go to the googleOuth20 link and then **google developers console** to get **GOOGLE\_CLIENT\_ID** and **GOOGLE\_CLIENT\_SECRET**

**process of crerating googleOauth apoplication:**

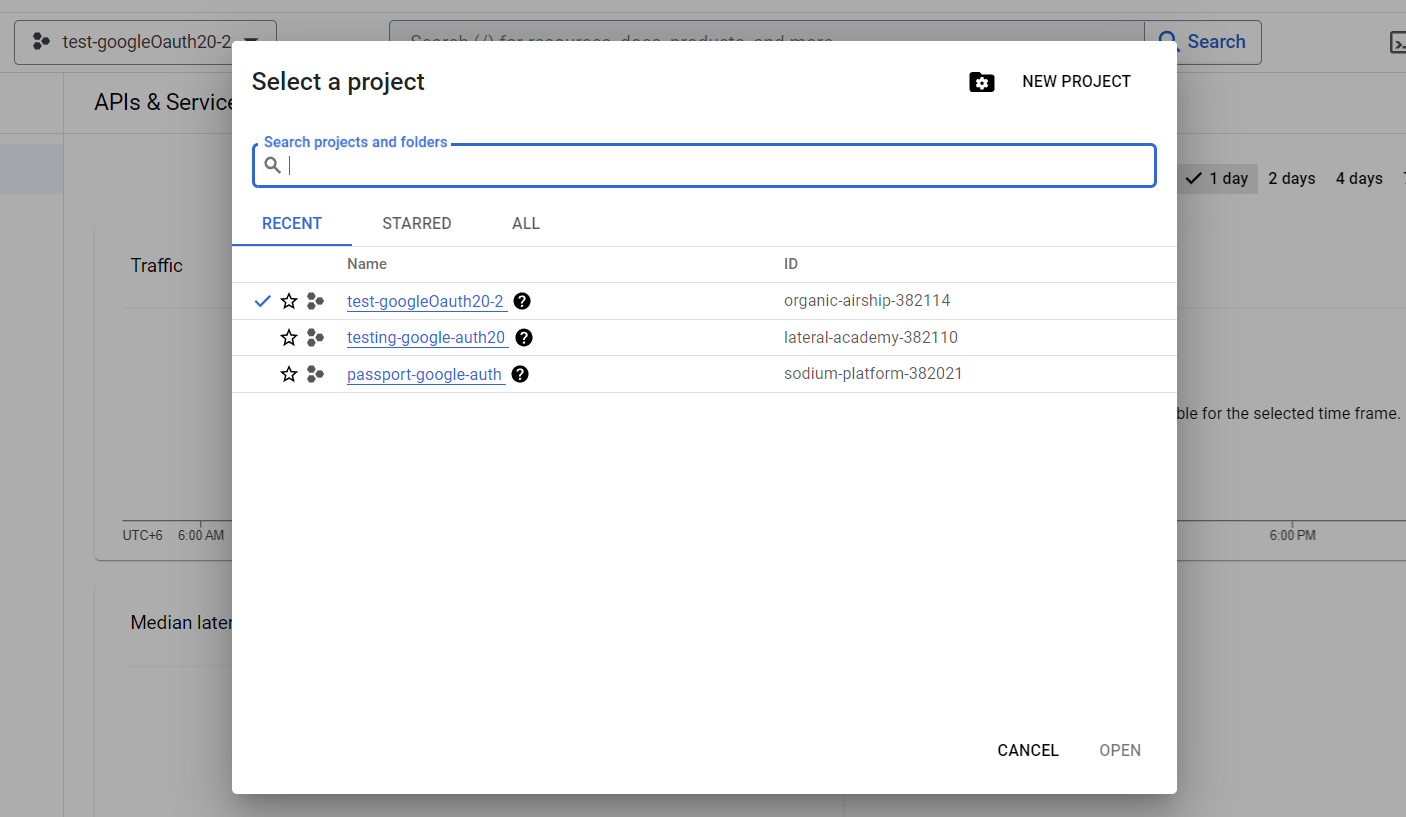
1. Go to ***google developer console***



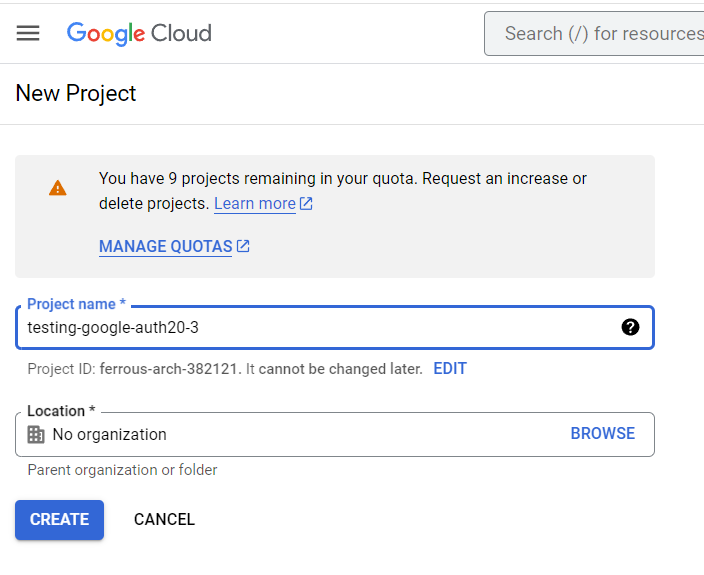


Click here to create new project

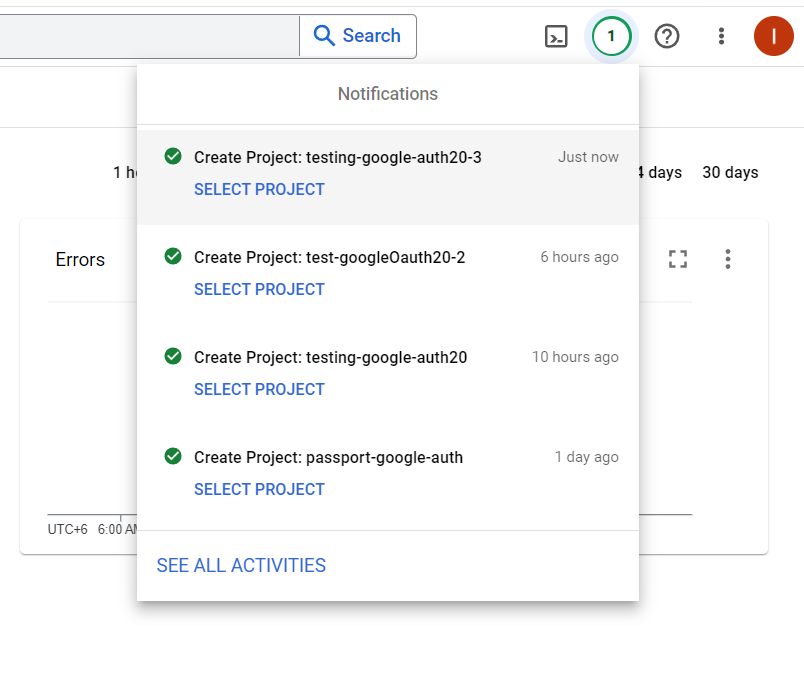
1. After clicking google developers console we can see the following page . then go to test-googleOauth20-2 to create a new project



Click here to create new project.

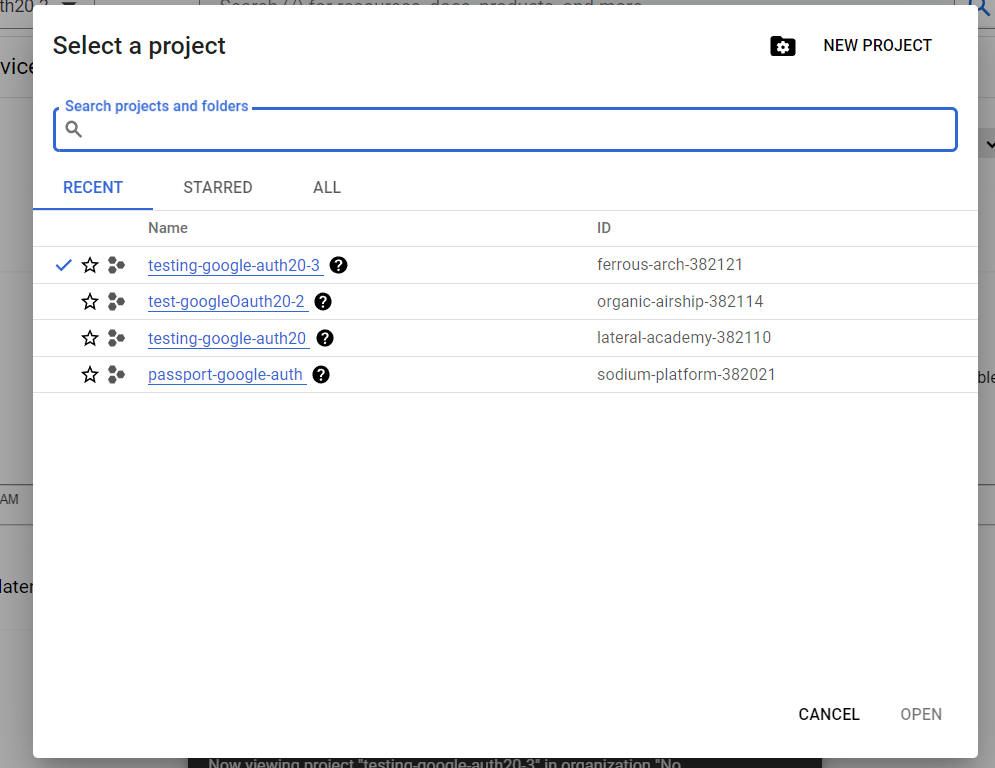
1. After clicking new project give a name of the project

Give a name of the project

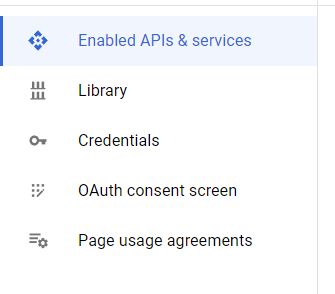
1. Then select the project 

Select the project

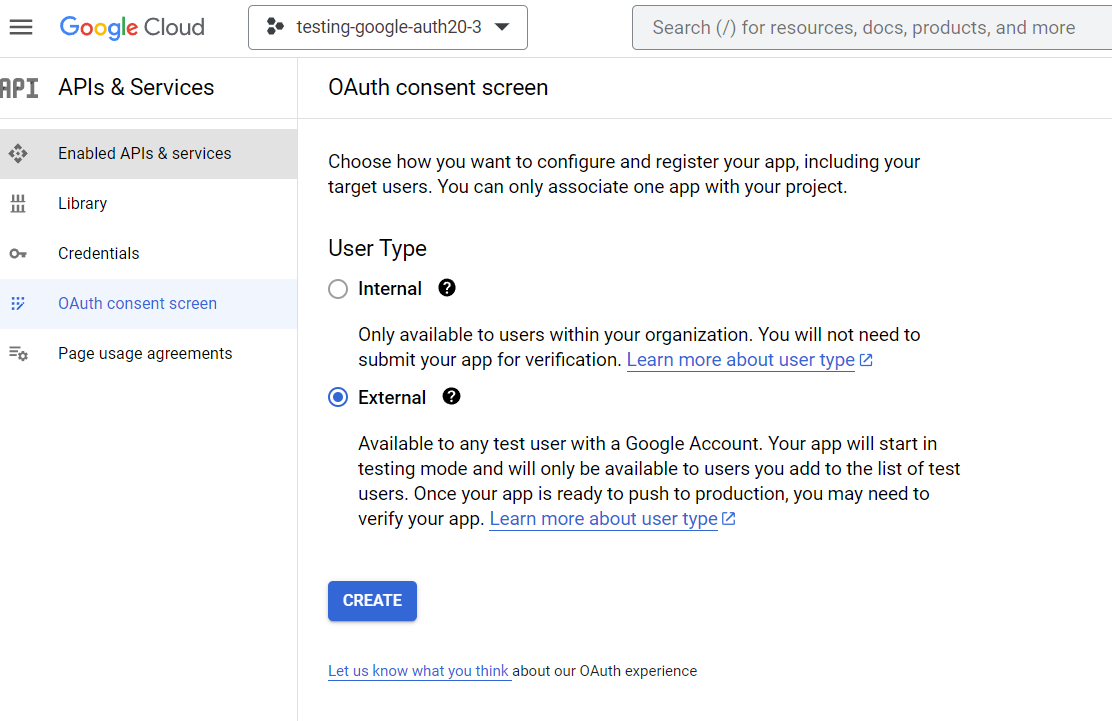
1. After that select the following project



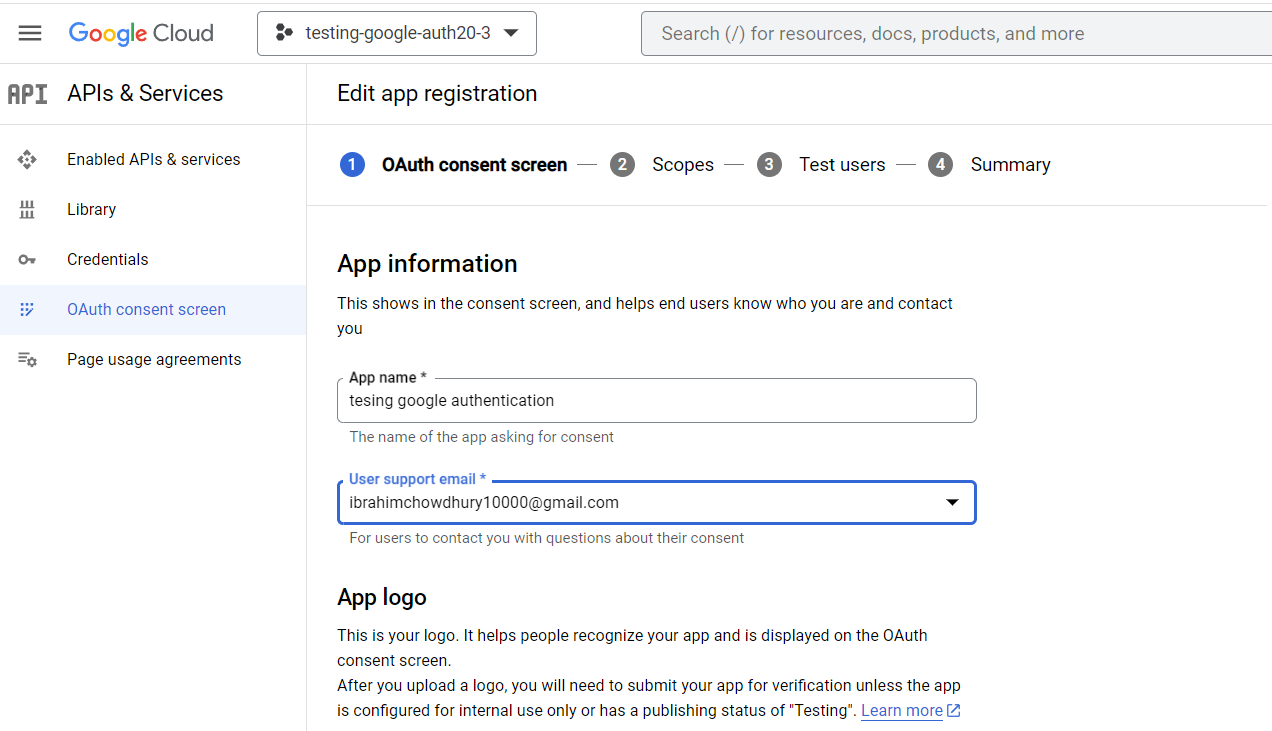
1. After that go to the OAuth consent screen



1. After clicking OAuth consent screen select the external and then create the project

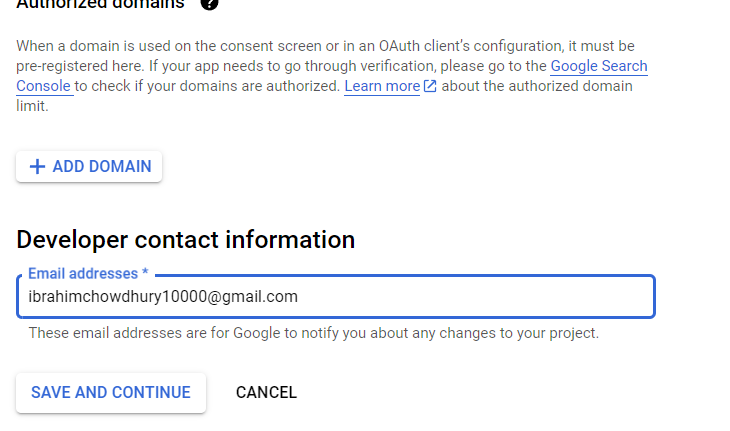


1. After that in the app information enter the app name and user supporter email

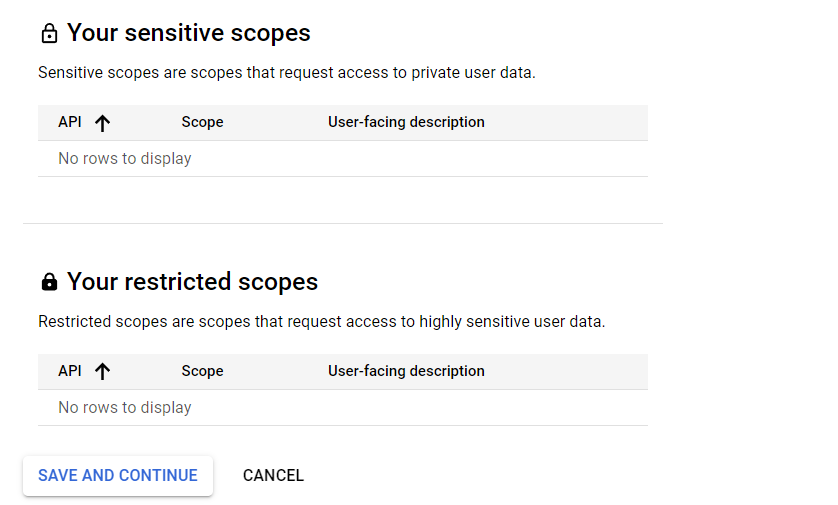


Enter app name and email

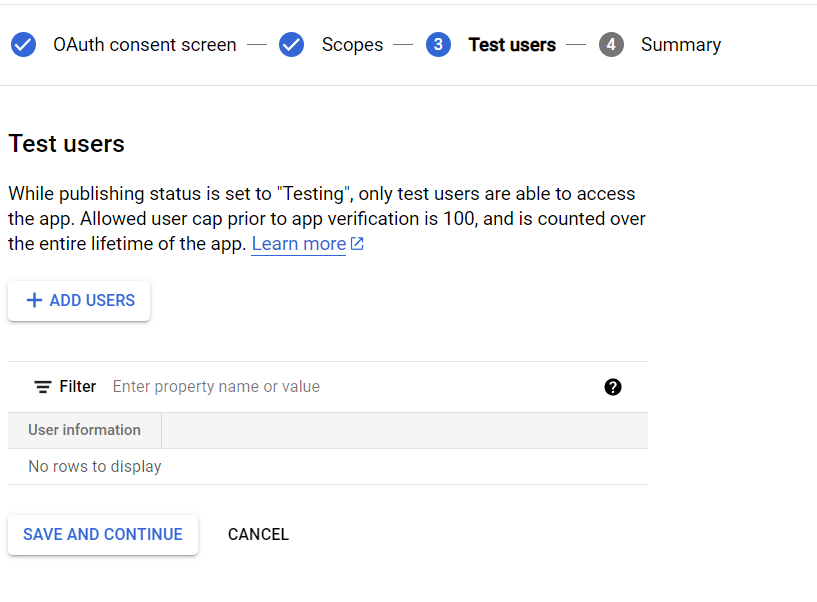
And then enter Developer contact information and then save and continue



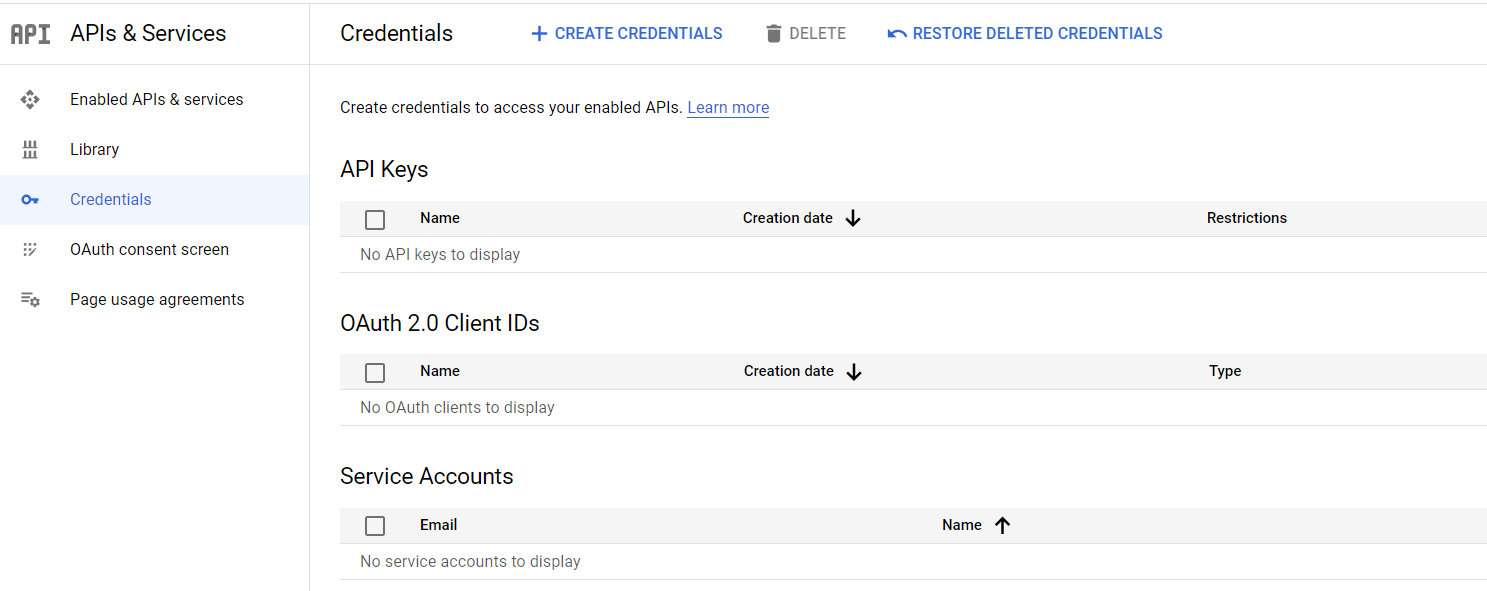
1. Then again save and continue



1. Again save and continue

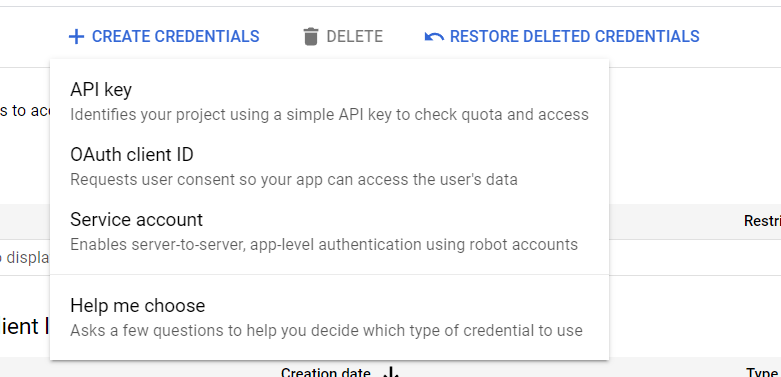


1. Then again click save and continue and then go to Credentials



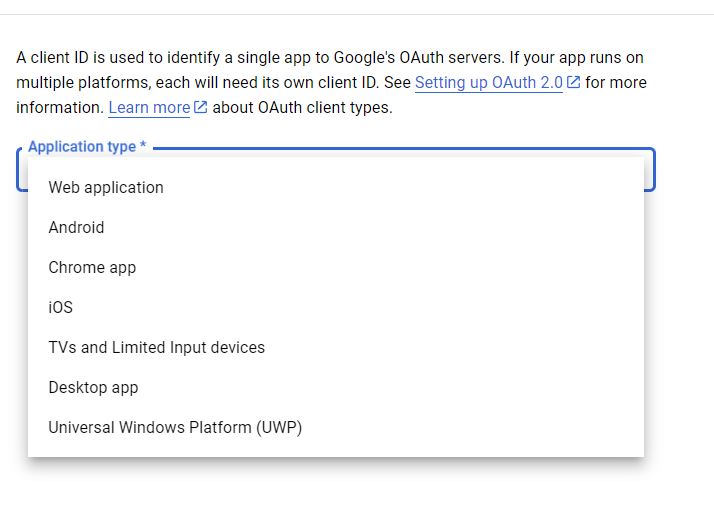
Click create credentials

1. After clicking the create credentials choose OAuth Client Id for getting google\_client\_id and google\_client\_secret

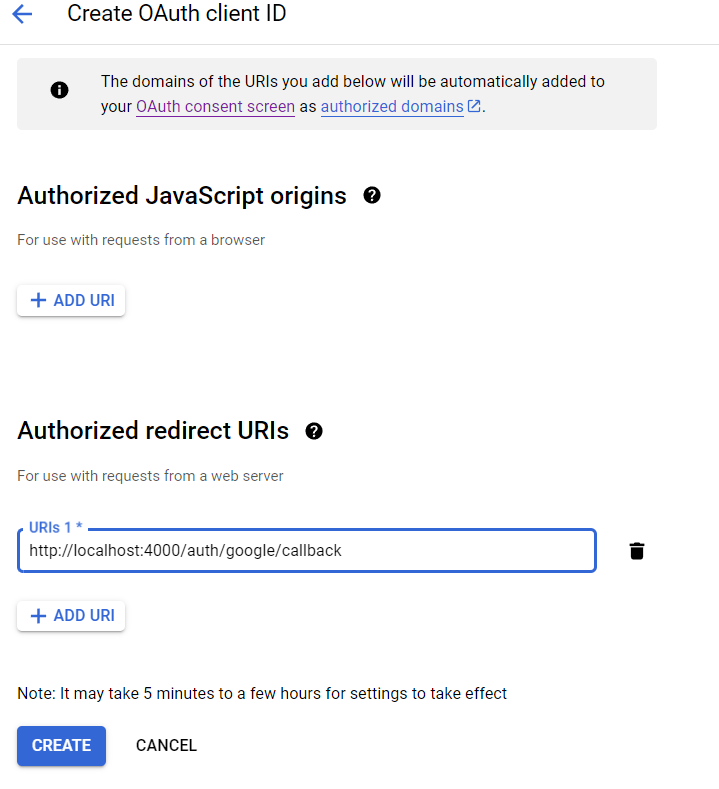


Click here to get id and secrect

1. After that select the application type to web application

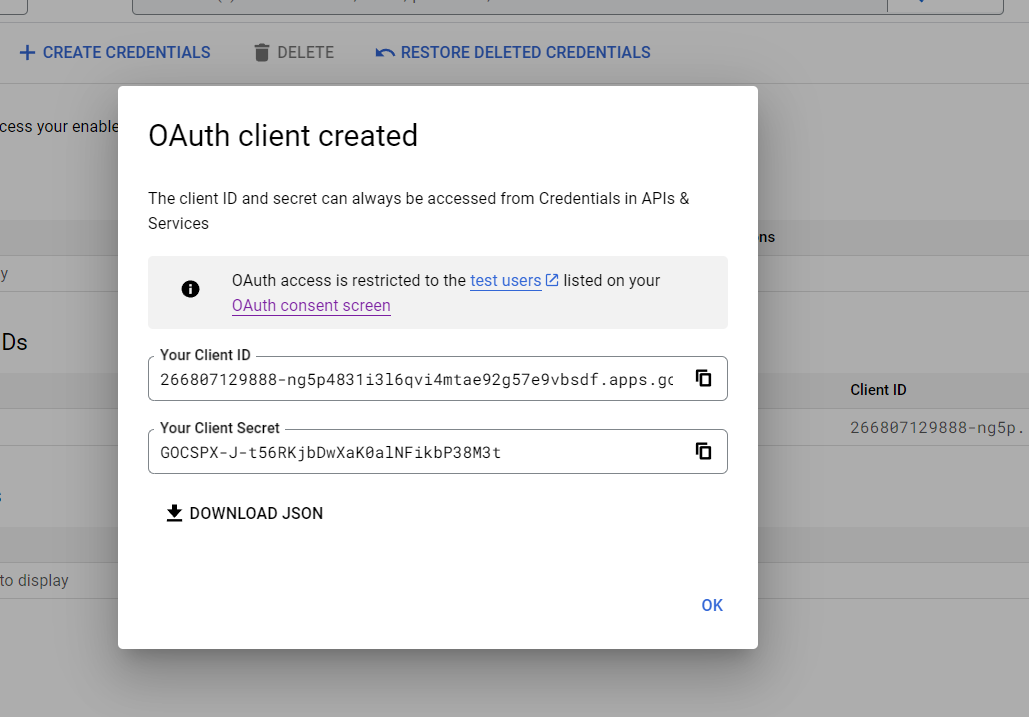


1. After selecting the web application add Authorize Redirect URLs and create the application



Here enter the callbackURL that we use in out application

1. After creating we get GOOGLE\_CLIENT\_ID and GOOGLE\_CLIENT\_SECRET



This client id and client secret we enter in the googleOauth20.js file

Then store these data in .env file

port=4000

DB\_url= mongodb://127.0.0.1:27017/google\_Oauth20

GOOGLE\_CLIENT\_ID=488978378899-kta5put2foae8dtid0prjb0q690eoj5s.apps.googleusercontent.com

GOOGLE\_CLIENT\_SECRET=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

That is how we can get id and secret of google\_client

**Using GoogleOauth20 strategy in our project :**

Requiring passport, googleStrategy, userModel and dotenv file

To know more about googleOauth20 go to the link <https://www.passportjs.org/packages/passport-google-oauth20/>

In googleOauth20.js enter the code bellow

let passport = require("passport")

let GoogleStrategy = require('passport-google-oauth20').Strategy;

require("dotenv").config()

let User = require("../userModel/usersModel")

passport.use(new GoogleStrategy({

    clientID: process.env.GOOGLE\_CLIENT\_ID,

    clientSecret: process.env.GOOGLE\_CLIENT\_SECRET,

    callbackURL: "http://localhost:4000/auth/google/callback"

Use this code for authenticate the user.

If the user is not found then create a new user and if the user is found then return the user

},

    async function (accessToken, refreshToken, profile, cb) {

        let user = await User.findOne({ googleId: profile.id })

        if (!user) {

            let newUser = User({

                username: profile.displayName,

                googleId: profile.id

            })

            await newUser.save()

            return cb(null, newUser)

        }

        else {

            return cb(null, user)

        }

    }

));

passport.serializeUser((user,done)=>{

    done(null, user.id)

})

passport.serializeUser() is setting id as cookie in user’s browser and passport.deserializeUser() is getting id from the cookie,

passport.deserializeUser(async(id, done)=>{

    let user= await User.findById(id)

    if(user)

    {

        done(null, user)

    }

})

After that write code below in app.js ,

let express=require("express")

let cors=require("cors")

let passport=require("passport")

let session=require("express-session")

let MongoStore=require("connect-mongo")

require("dotenv").config()

require("./passport/googleOauth20")

let app=express()

app.use(cors())

app.set('trust proxy', 1) // trust first proxy

app.use(session({

  secret: 'keyboard cat',

  resave: false,

  saveUninitialized: true,

  store: MongoStore.create({

    mongoUrl:process.env.DB\_url,

    collectionName:"sessions"

  })

}))

app.use(passport.initialize())

app.use(passport.session())

app.get("/",(req,res)=>{

   res.sendFile(\_\_dirname + "/view/home.html")

})

Check if the req is authenticated or not. If authenticated then go to profile page and if not authenticate then go to login page

let isLoggedin=(req,res,next)=>{

    if(req.isAuthenticated())

    {

        return res.redirect("/profile")

    }

    return next()

}

app.get("/login", isLoggedin, (req,res)=>{

       res.sendFile(\_\_dirname + "/view/login.html")

})

Put this code from googleOauth20 site to authenticate the user. If the user is not authenticate then go to login page and if the user is authenticate then go to profile page

app.get('/auth/google',

  passport.authenticate('google', { scope: ['profile'] }));

app.get('/auth/google/callback',

  passport.authenticate('google', { failureRedirect: '/login',

successRedirect:"/profile" }),

  function(req, res) {

    // Successful authentication, redirect home.

    res.redirect('/');

  });

let checkIsLogged=(req,res,next)=>{

    if(req.isAuthenticated()){

       return next()

    }

    return res.redirect("/login")

}

app.get("/profile", checkIsLogged, (req,res)=>{

        res.sendFile(\_\_dirname + "/view/profile.html")

})

Log out request to log out of the user.

app.get("/logout",(req,res)=>{

    req.logOut(()=>{

        res.redirect("/")

    })

})

module.exports=app