Topic: Creating Plant Analysis Tools Using Gemini AI and Express.js

Speaker: | Notebook: Django Project: Car Listing

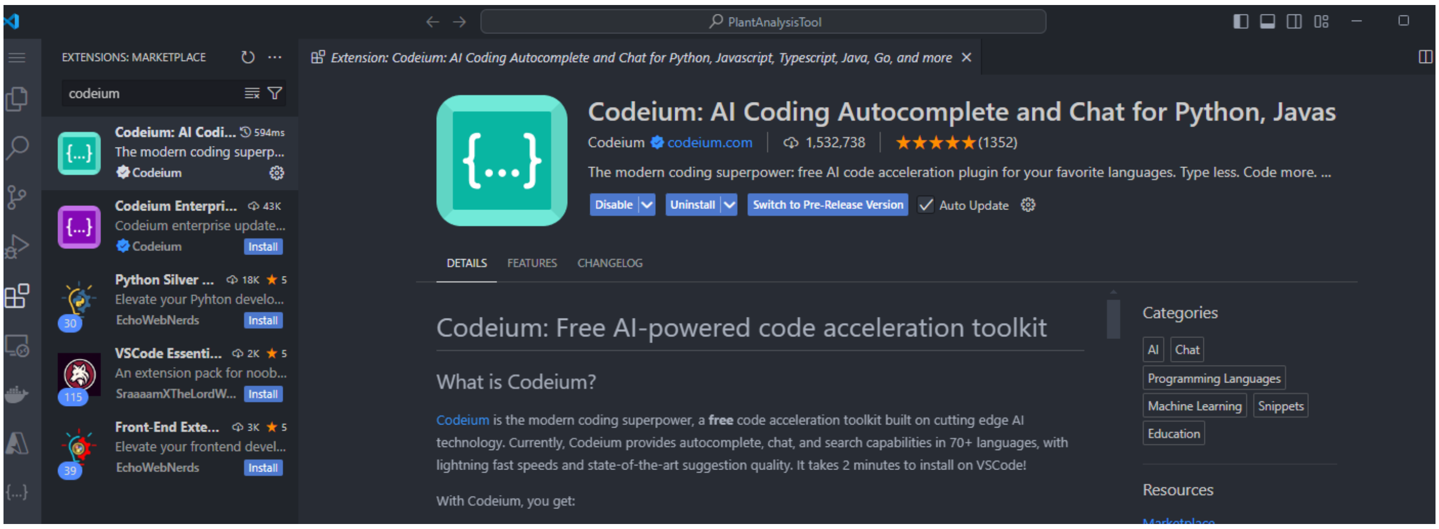
We installed the NODE.JS (Javascript) for this project. You need to download from here.

Node.js for Javascript and Django for Python are 2 different platforms.

Project's Main Resource Page: YouTube video here.

1. Open a new folder, PLANTANALYSISTOOL. Open a new Git bash terminal, and we issue the command, CODE . (code dot) to open the folder in VS Code.

We install the extension CODEIUM AI.



PlantAnalysisTool

EXTENSIONS: MARKETPLACE

codeium

Codeium: AI Codi... 594ms
The modern coding superp...
Codeium

Codeium Enterpri... 43K
Codeium enterprise update...
Codeium Install

Python Silver ... 18K ★ 5
Elevate your Pyhton develo...
EchoWebNerds Install

VSCode Essenti... 2K ★ 5
An extension pack for noob...
SraaaamXTheLordW... Install

Front-End Exte... 3K ★ 5
Elevate your frontend devel...
EchoWebNerds Install

Extension: Codeium: AI Coding Autocomplete and Chat for Python, Javascript, Typescript, Java, Go, and more

Codeium: AI Coding Autocomplete and Chat for Python, Javas

Codeium codeium.com | 1,532,738 | ★★★★★(1352)

The modern coding superpower: free AI code acceleration plugin for your favorite languages. Type less. Code more. ...

Disable | Uninstall | Switch to Pre-Release Version | ✓ Auto Update

DETAILS FEATURES CHANGELOG

Codeium: Free AI-powered code acceleration toolkit

What is Codeium?

Codeium is the modern coding superpower, a **free** code acceleration toolkit built on cutting edge AI technology. Currently, Codeium provides autocomplete, chat, and search capabilities in 70+ languages, with lightning fast speeds and state-of-the-art suggestion quality. It takes 2 minutes to install on VSCode!

With Codeium, you get:

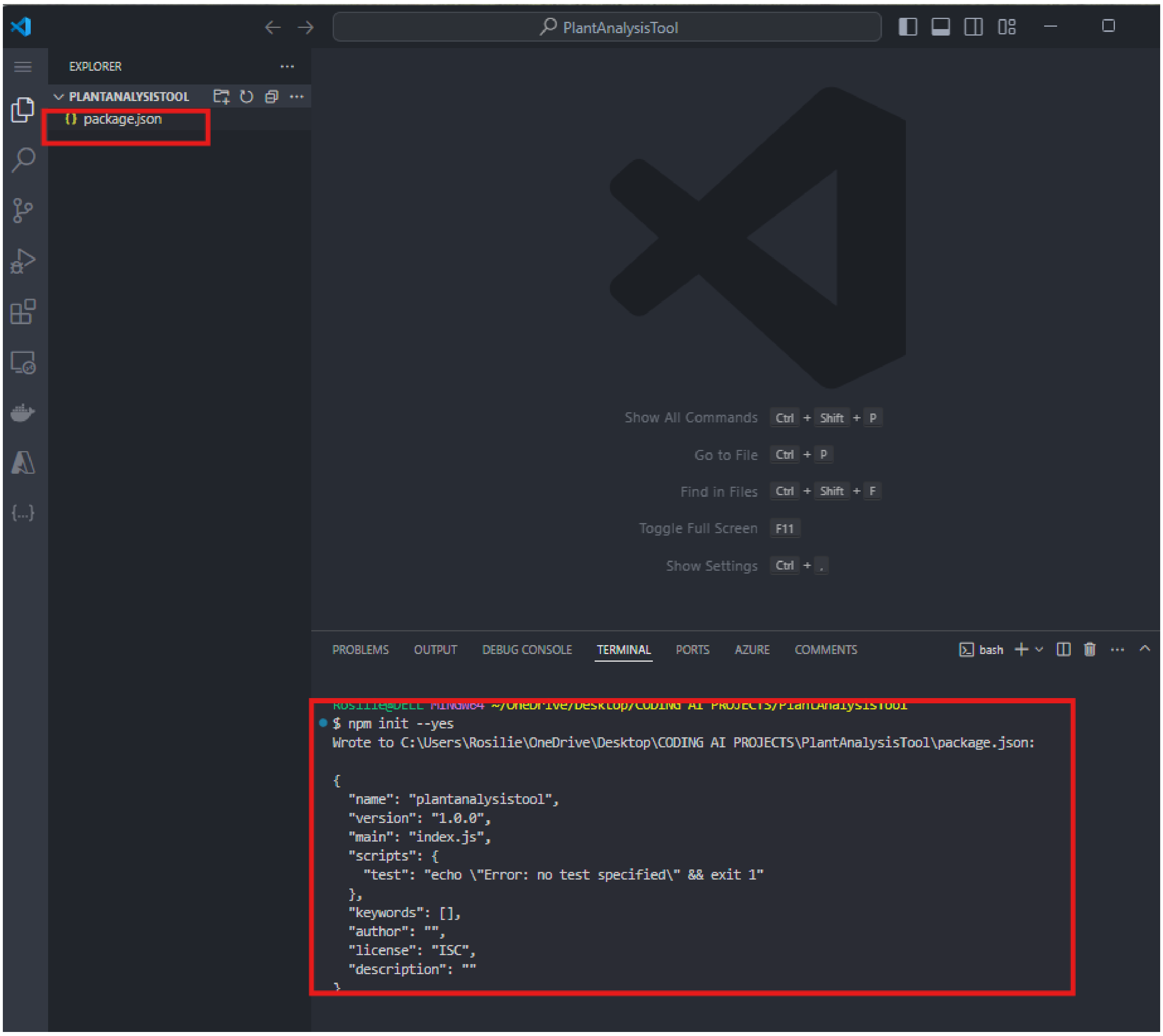
Categories

AI | Chat | Programming Languages | Machine Learning | Snippets | Education

Resources

2. We issue the command for Node.js the NPM (which is PIP in Django) to create the PACKAGE.JSON file.

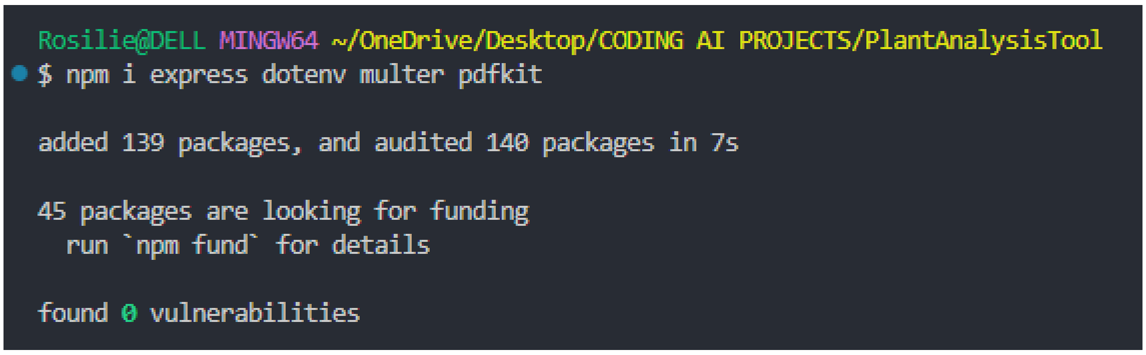
```
$ npm init --yes
```



```
Rosilie@DELL MINGW64 ~/OneDrive/Desktop/CODING AI PROJECTS/PlantAnalysisTool
$ npm init --yes
Wrote to C:\Users\Rosilie\OneDrive\Desktop\CODING AI PROJECTS\PlantAnalysisTool\package.json:

{
  "name": "plantanalysistool",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

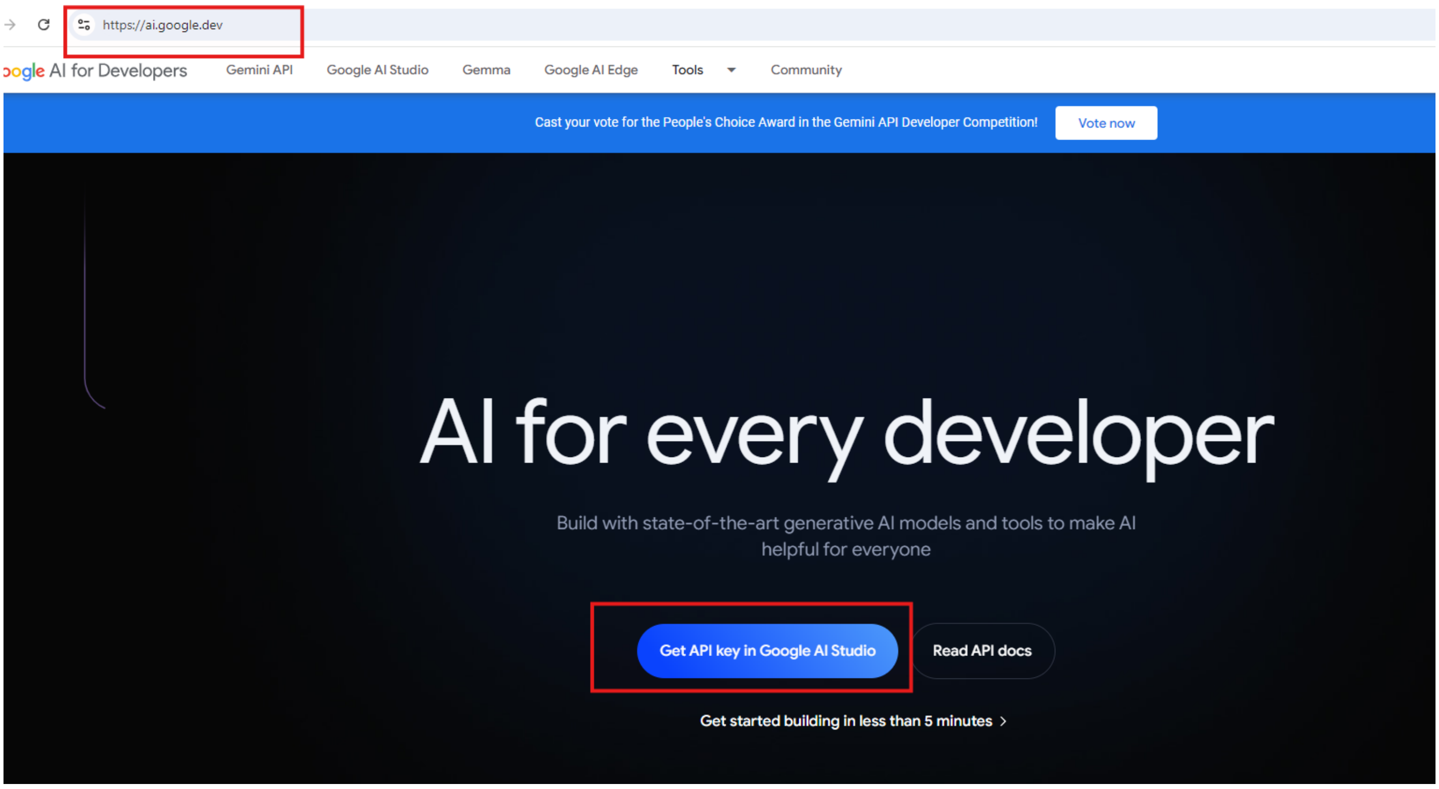
3. We install the packages we need.

```
$ npm i express dotenv multer pdfkit
```



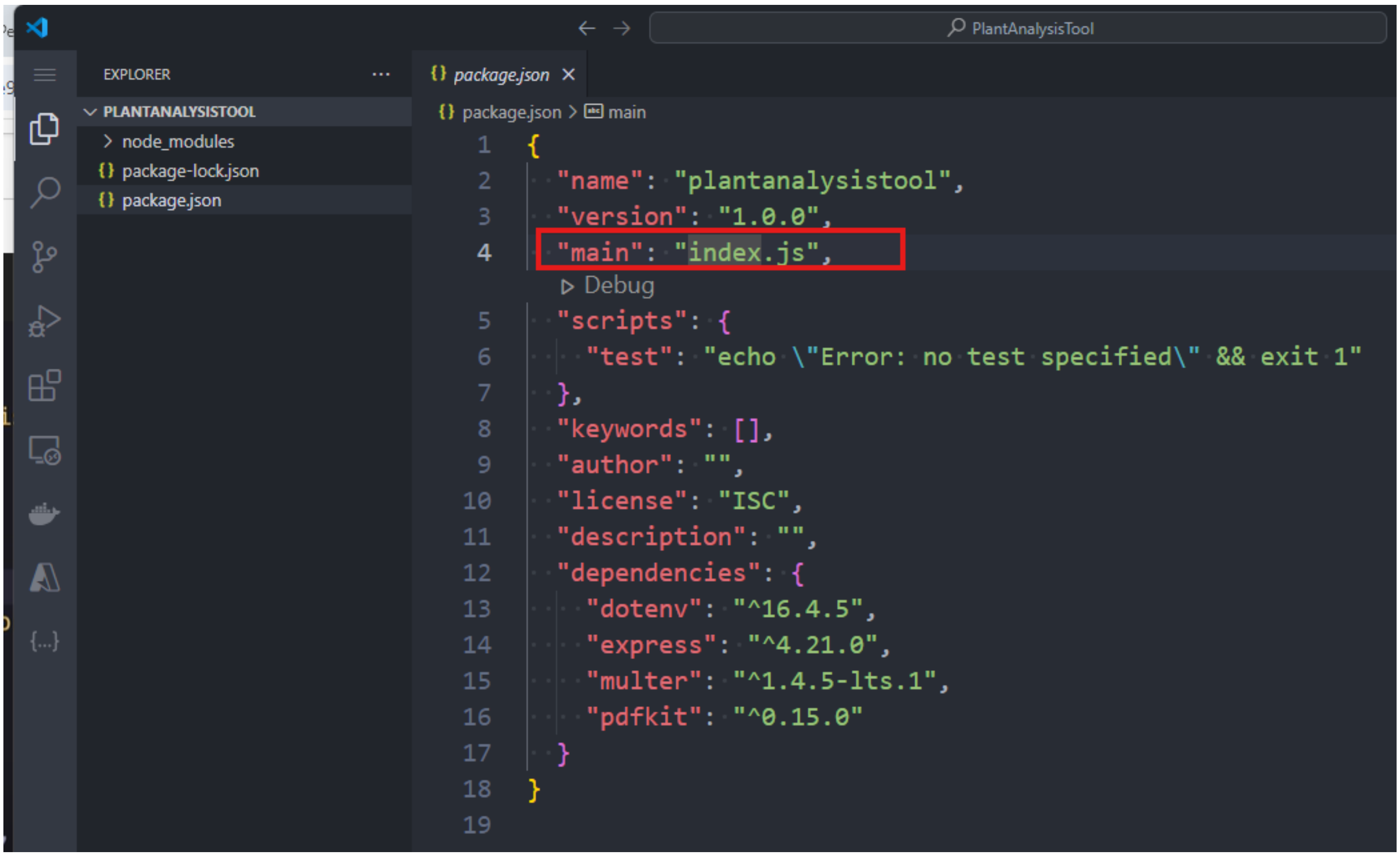
```
Rosilie@DELL MINGW64 ~/OneDrive/Desktop/CODING AI PROJECTS/PlantAnalysisTool
$ npm i express dotenv multer pdfkit

added 139 packages, and audited 140 packages in 7s

45 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

4. Get GEMINI API KEY. Simply in the Google search, type GEMINI API.


https://ai.google.dev
Google AI for Developers
Gemini API
Google AI Studio
Gemma
Google AI Edge
Tools
Community
Cast your vote for the People's Choice Award in the Gemini API Developer Competition!
Vote now
AI for every developer
Build with state-of-the-art generative AI models and tools to make AI helpful for everyone
Get API key in Google AI Studio
Read API docs
Get started building in less than 5 minutes

5. Update the PACKAGE.JSON:

FROM:

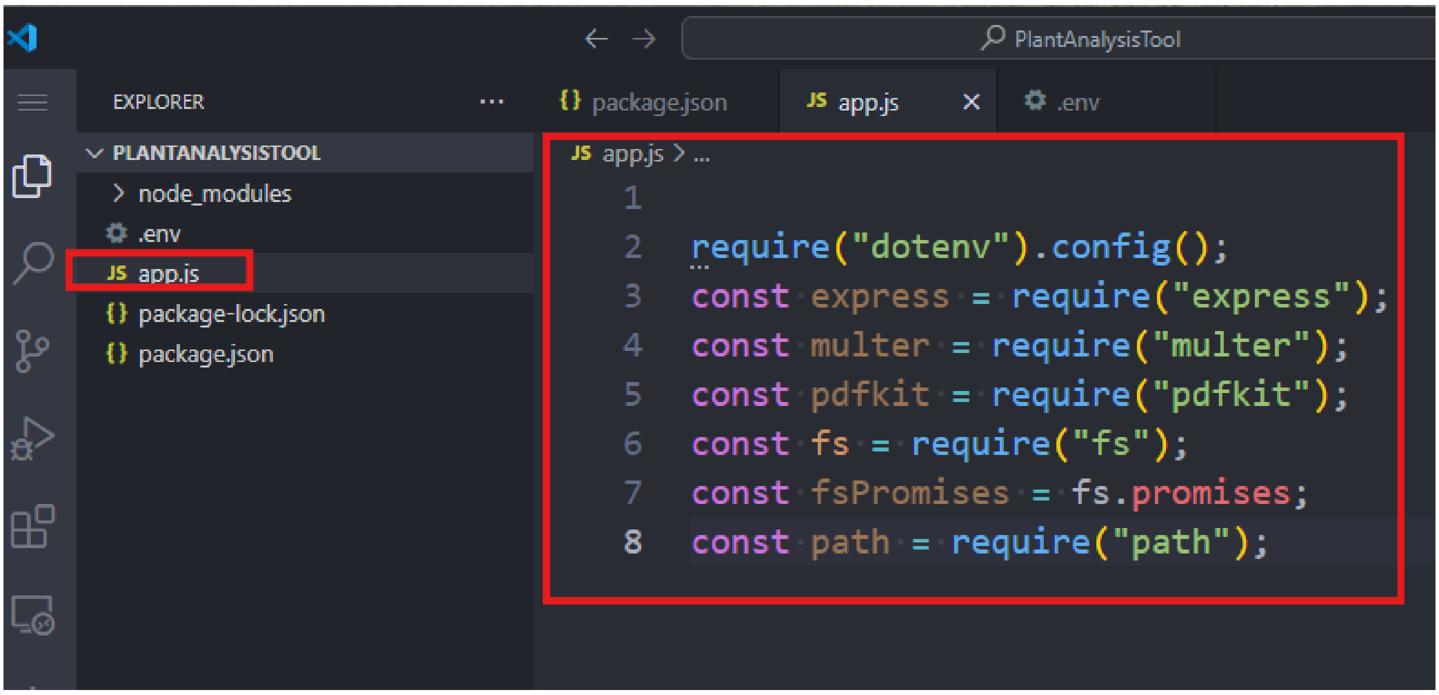


```
{
  "name": "plantanalysistool",
  "version": "1.0.0",
  "main": "index.js",
  ▷ Debug
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "dotenv": "^16.4.5",
    "express": "^4.21.0",
    "multer": "^1.4.5-lts.1",
    "pdfkit": "^0.15.0"
  }
}
```

TO:

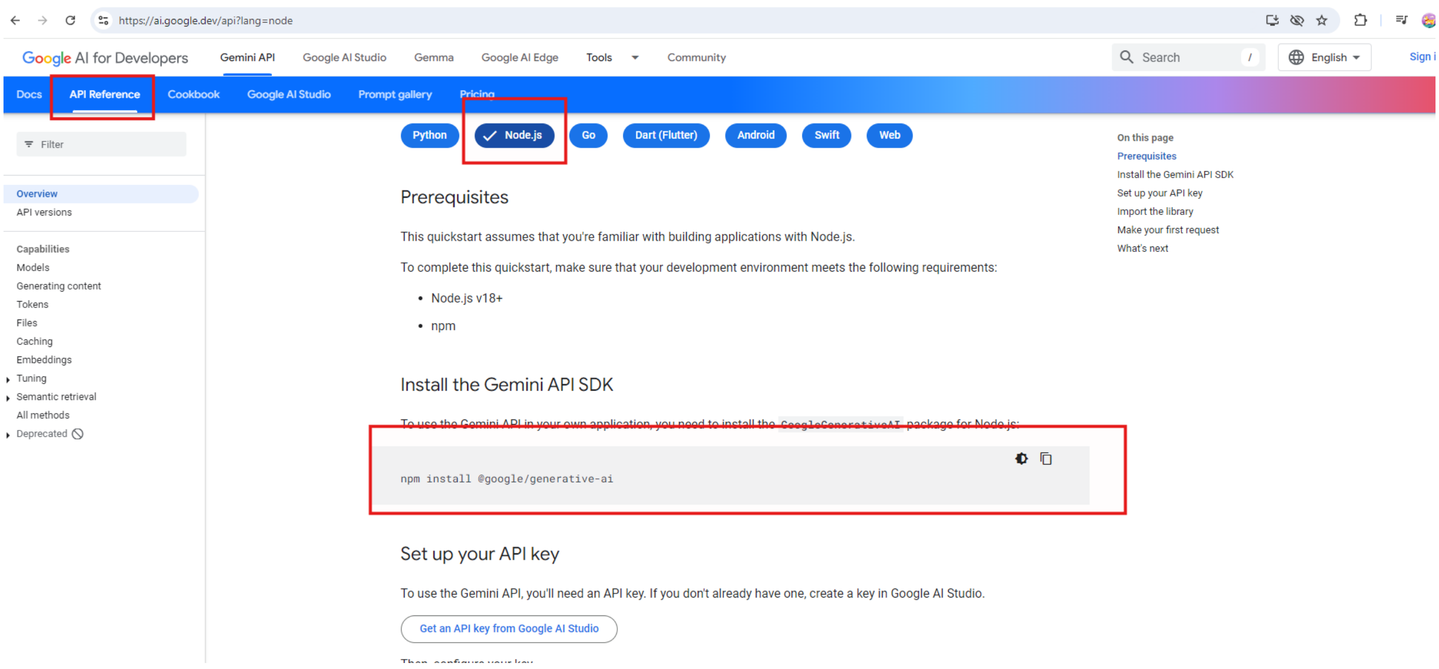
6.We created files APP.JS and .ENV files in the root directory.

In the APPS.JS:



```
require("dotenv").config();
const express = require("express");
const multer = require("multer");
const pdfkit = require("pdfkit");
const fs = require("fs");
const fsPromises = fs.promises;
const path = require("path");
```

7. On Google AI Dashboard, select API REFERENCE, choose NODE.JS and get the code to install GEMINI API



Prerequisites

This quickstart assumes that you're familiar with building applications with Node.js.

To complete this quickstart, make sure that your development environment meets the following requirements:

- Node.js v18+
- npm

Install the Gemini API SDK

To use the Gemini API in your own application, you need to install the `GoogleGenerativeAI` package for Node.js:

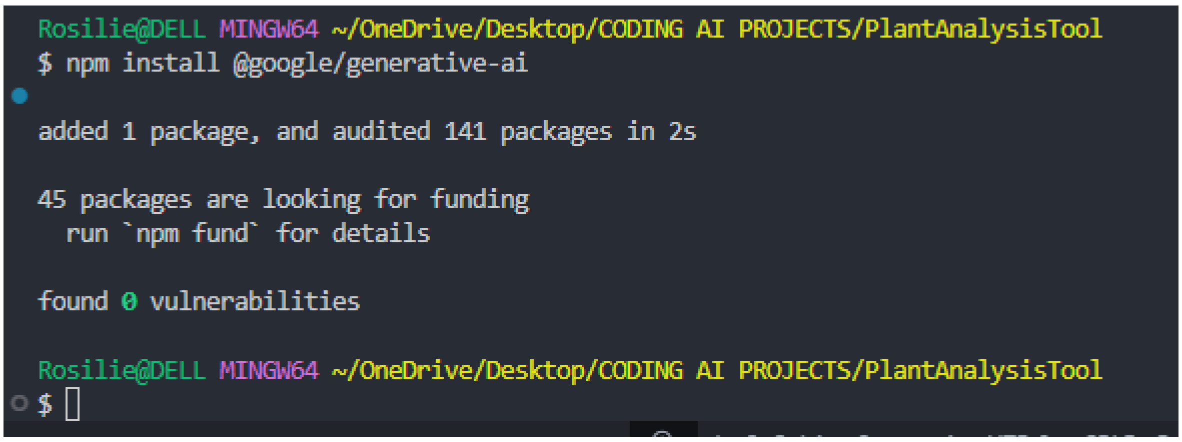
```
npm install @google/generative-ai
```

Set up your API key

To use the Gemini API, you'll need an API key. If you don't already have one, create a key in Google AI Studio.

8. Install the GEMINI API SDK in the terminal:

```
$ npm install @google/generative-ai
```



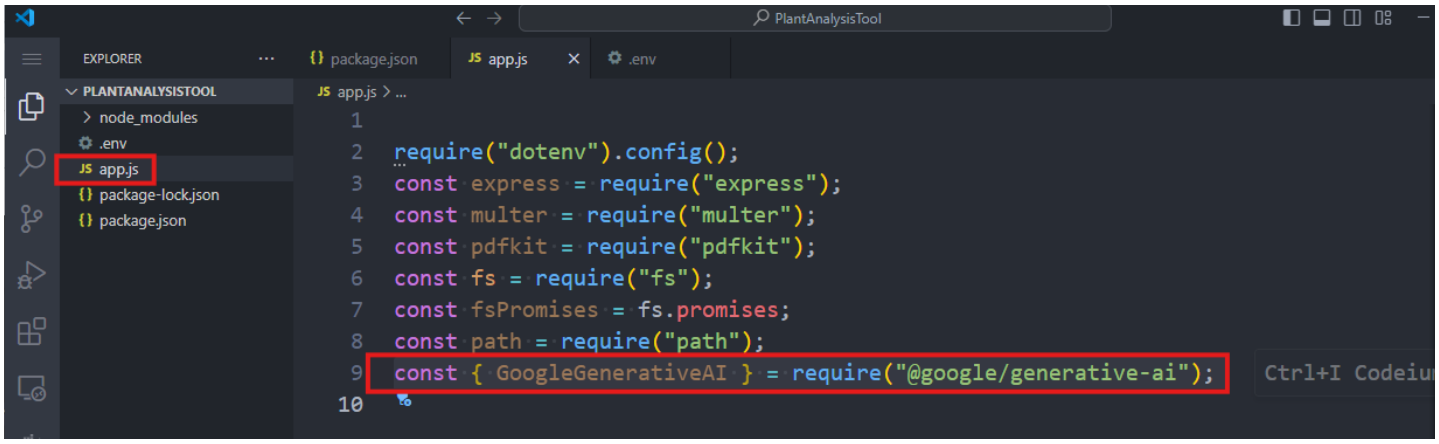
```
Rosilie@DELL MINGW64 ~/OneDrive/Desktop/CODING AI PROJECTS/PlantAnalysisTool
$ npm install @google/generative-ai

added 1 package, and audited 141 packages in 2s

45 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

Rosilie@DELL MINGW64 ~/OneDrive/Desktop/CODING AI PROJECTS/PlantAnalysisTool
$ 
```

9. We copy this from the GEMINI API REFERENCE into our APPS.JS

Import the library

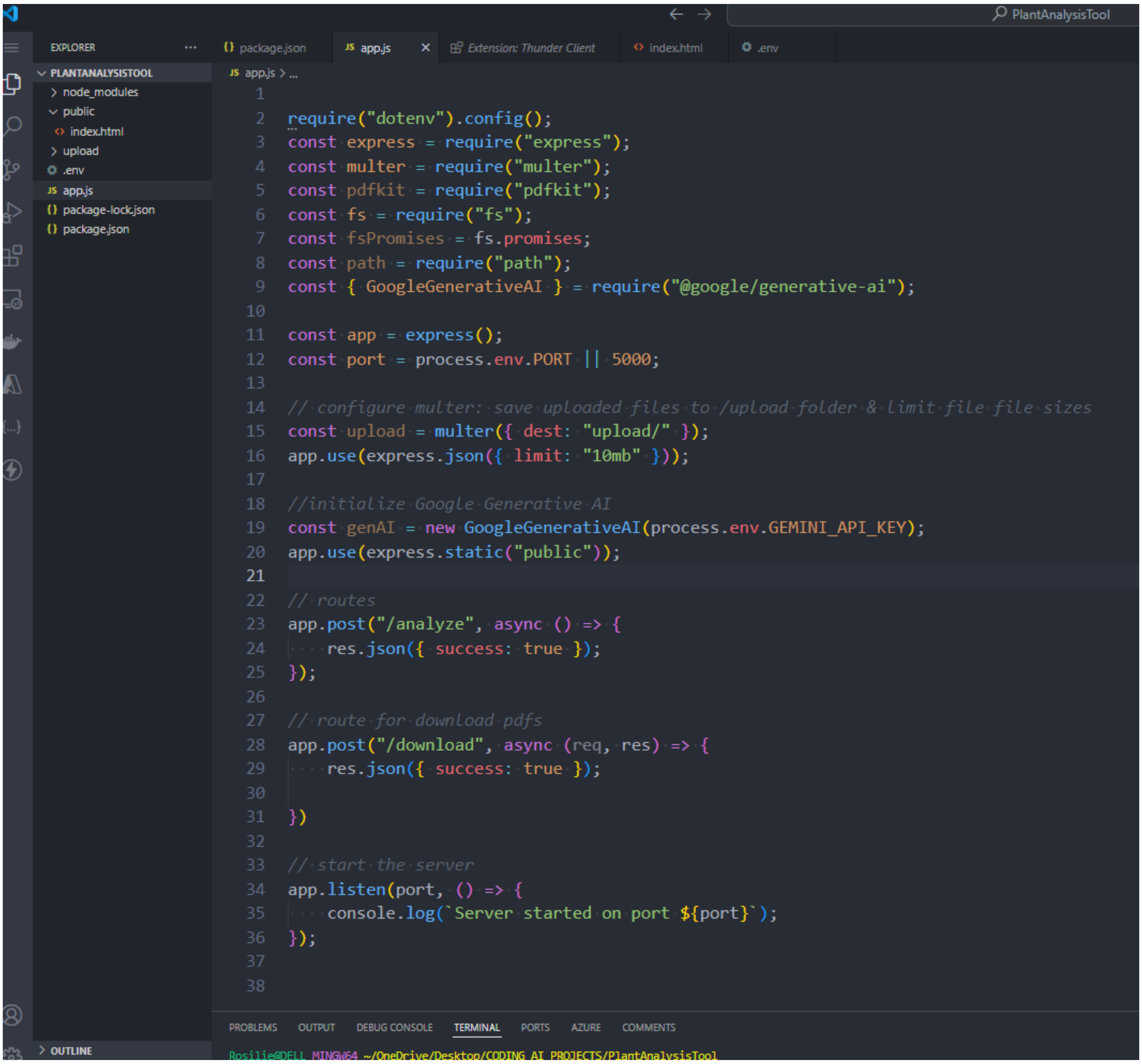
Import the Google Generative AI library.

```
const { GoogleGenerativeAI } = require("@google/generative-ai");
```



```
require("dotenv").config();
const express = require("express");
const multer = require("multer");
const pdfkit = require("pdfkit");
const fs = require("fs");
const fsPromises = fs.promises;
const path = require("path");
const { GoogleGenerativeAI } = require("@google/generative-ai");
```

10. We updated our APPS.JS AS:

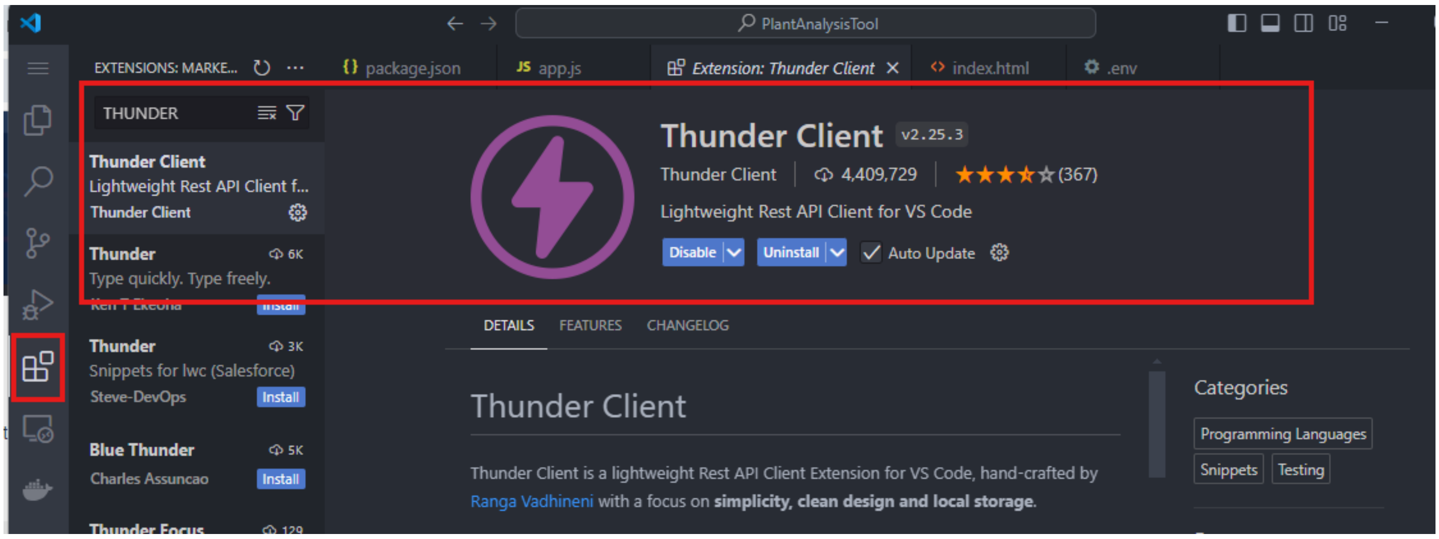


```
JS app.js > ...

require("dotenv").config();
const express = require("express");
const multer = require("multer");
const pdfkit = require("pdfkit");
const fs = require("fs");
const fsPromises = fs.promises;
const path = require("path");
const { GoogleGenerativeAI } = require("@google/generative-ai");

const app = express();
const port = process.env.PORT || 5000;

// configure multer: save uploaded files to /upload folder & limit file file sizes
const upload = multer({ dest: "upload/" });
app.use(express.json({ limit: "10mb" }));

//initialize Google Generative AI
const genAI = new GoogleGenerativeAI(process.env.GEMINI_API_KEY);
app.use(express.static("public"));

// routes
app.post("/analyze", async () => {
    res.json({ success: true });
});

// route for download pdfs
app.post("/download", async (req, res) => {
    res.json({ success: true });

})

// start the server
app.listen(port, () => {
    console.log(`Server started on port ${port}`);
});

```

```
Rosilie@DELL MINGW64 ~/OneDrive/Desktop/CODING AI PROJECTS/PlantAnalysisTool
```

11. To test our work, run NODE --WATCH APP (where app is our APP.JS)

Then use POSTMAN (or INSOMNIA) or add an extension THUNDER CLIENT as a VSCODE extension to test the ENDPOINT WITHIN VS CODE.



Thunder Client v2.25.3

Thunder Client | 4,409,729 | ★★★★☆ (367)

Lightweight Rest API Client for VS Code

Disable | Uninstall | Auto Update

DETAILS FEATURES CHANGELOG

Thunder Client

Thunder Client is a lightweight Rest API Client Extension for VS Code, hand-crafted by Ranga Vadhineni with a focus on **simplicity, clean design and local storage.**

Categories: Programming Languages, Snippets, Testing

12.