Configuration Manual

Ragul Murugesan
Student ID: 23154870

1. Introduction

This document provides detailed instruction about the components that are needed, and respective configuration that needs to be done in order to recreate the project environment in another machine. The project has to install node modules that helps to run the web application. This can we run on any platform as long as node runtime is installed with its respective packages. Detailed follow through installations will be continued below.

2. Hardware Machine Configuration

The physical machine on which setup was done is a Techno Megabook T1 with the following specifications.

System Configuration	
Operating System	Microsoft Windows 11 Home Single Edition 64bit
Processor	Intel i7 11th Gen – 1165G7 @ 2.80GHz, Logical Procssor:8, cores: 4
Ram	16GB
Virtual Memory	9.45GB
Graphics Card	4GB
Virtualization	Enabled at Bios

3. Software requirements:

The upcoming stages will be discussed about the installation and configuration of software components needed for this project.

- NodeJs: v23.3.0[1]

  Node.js is not dependent on any operating system software. It can work on Linux, macOS, or Windows. Node.js is a JavaScript runtime environment with community packages with unique functionality respectively.

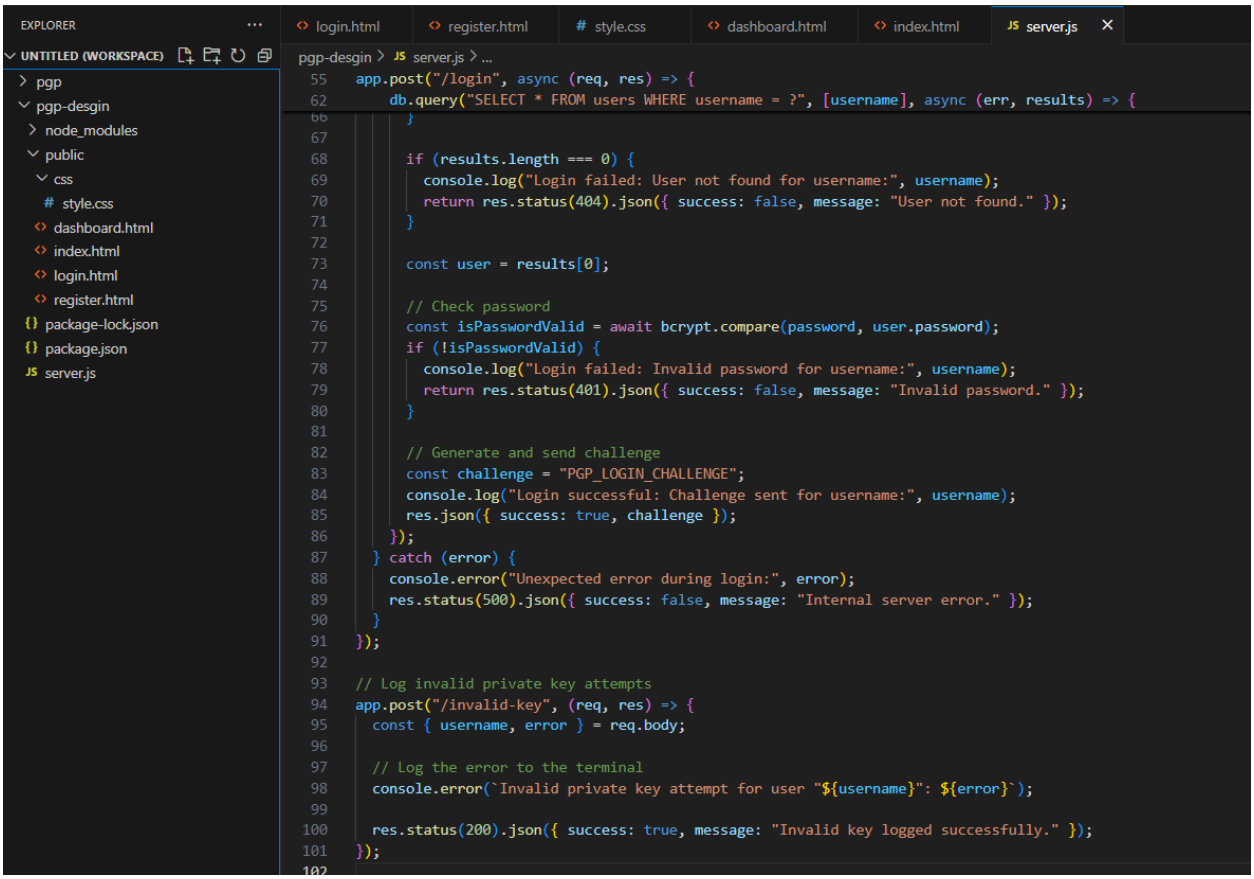
  NodeJs installation guide from their respective official website.

- Google Chrome: v 133.0.6876.4

  Browser to view the running application with url: localhost.
- Visual Studio Code: v 1.95.3[2]

  Code editor to view source code to edits, implements and updates.
- MySQL: v 8.0.41[3]

Used to store data of user information and public key when registration.



3.1 Javascript, with servercode in Visual Studio code IDE.

4. Installed Package Dependencies:

The below mentioned packages are basic ones to get the application running and its mandatory so it can be installed “npm install bcryptjs body-parser express mysql2 openpgp” or just “npm install” will install the dependencies from projects package.json.

- bcryptjs: ^2.4.3
- body-parser: ^1.20.3
- express: ^4.21.1
- mysql2: ^3.11.3
- openpgp: ^5.11.2[4]

```
const express = require("express");
const bcrypt = require("bcryptjs");
const mysql = require("mysql2");
const bodyParser = require("body-parser");
const path = require("path");
const openpgp = require("openpgp");
```

4.1 Importing packages into server.js

5. Connecting SeverJS to Mysql Database:

Step1: Need to change values of user, password, database(schema) to your needs.

```
// Set up MySQL database connection
const db = mysql.createConnection({
    host: "localhost",
    user: "root",
    password: "root",
    database: "pgp"
});
```

5.1 Change the user, password, database information

Step2: create Database and Tables with below query.

CREATE SCHEMA pgp;

CREATE TABLE pgp.users (

username VARCHAR(255) NOT NULL,

email VARCHAR(255) NOT NULL UNIQUE,

password VARCHAR(255) NOT NULL,

public_key TEXT NOT NULL,

PRIMARY KEY (username)

);

6. Execution and Testing:

Execution and testing phase consists of the following

Phase1: Run “node server.js” to get the application running then enter localhost:3000 on chrome browser.

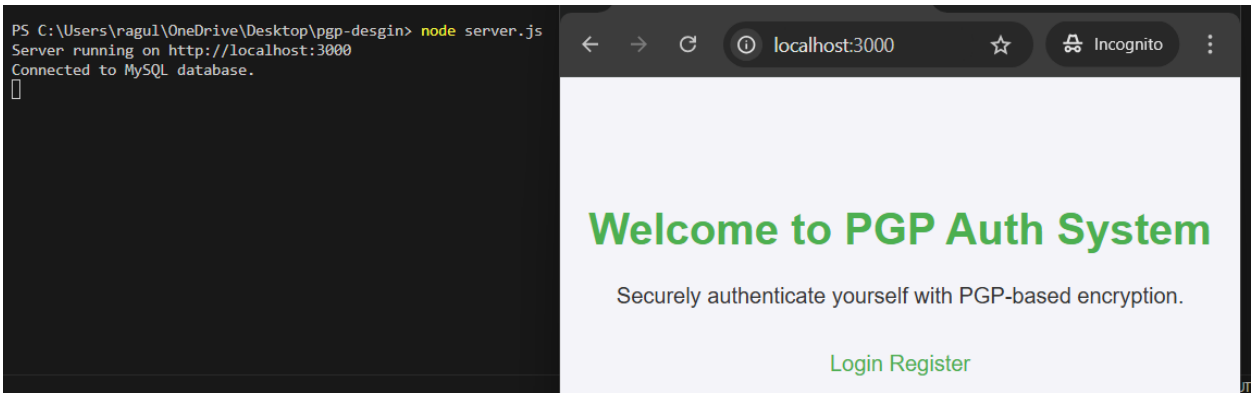


Figure 6.1 Start Sever

Phase2: Registration Page, user will enter the following details shown in the browser once that's done click register button and a unique private_key will get downloaded in the users system.

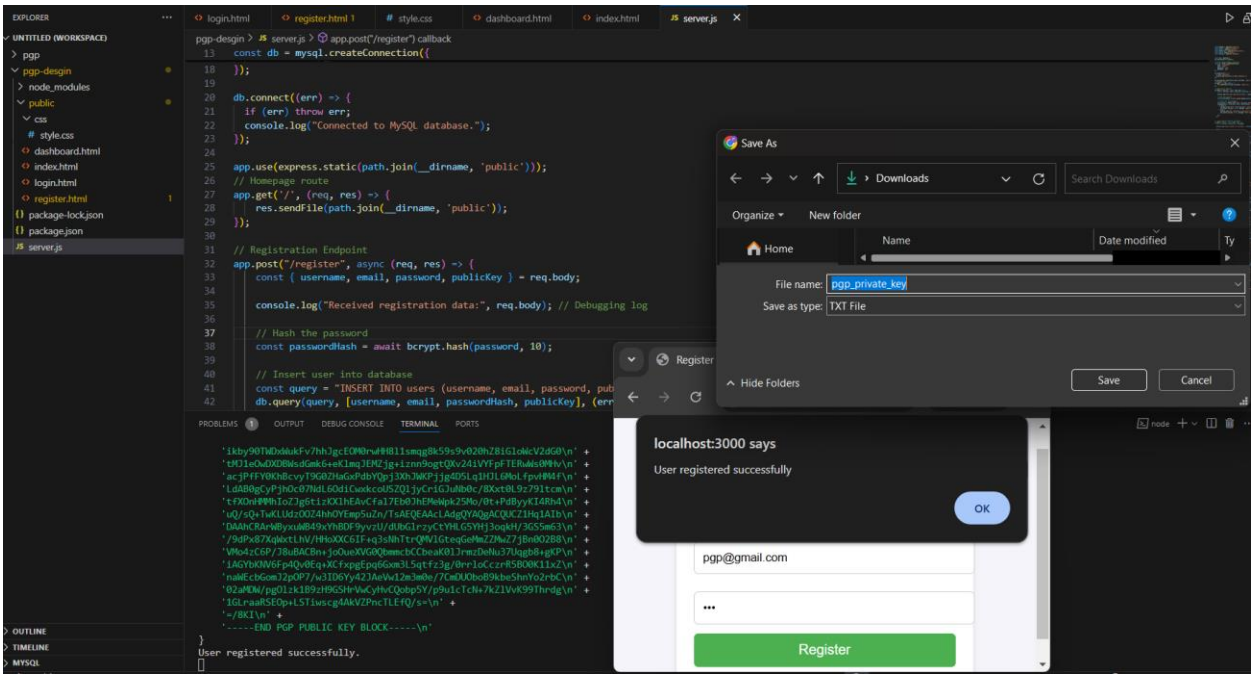
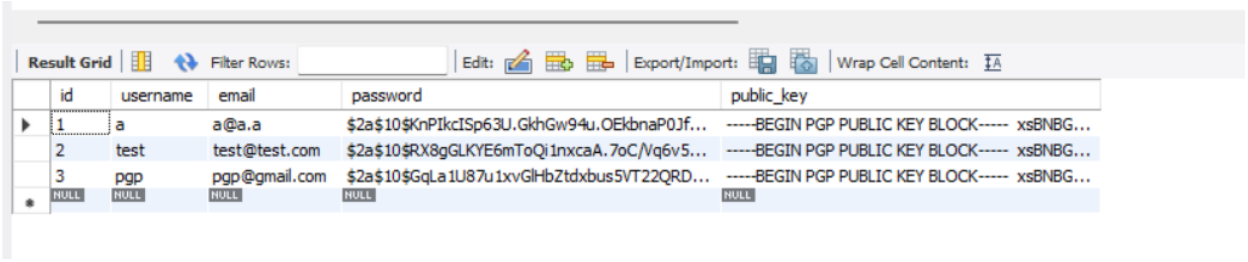


Figure 6.2 User registration and Private_Key Generation



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	username	email	password	public_key
1	a	a@a.a	$2a$10$KnPIkcISp63U.GkhGw94u.OEkbnaP0Jf...	-----BEGIN PGP PUBLIC KEY BLOCK----- xsBNBG...
2	test	test@test.com	$2a$10$RX8gGLKYE6mToQi1nxcaA.7oC/Vq6v5...	-----BEGIN PGP PUBLIC KEY BLOCK----- xsBNBG...
3	pgp	pgp@gmail.com	$2a$10$GqLa1U87u1xvGlHbZtdxbus5VT22QRD...	-----BEGIN PGP PUBLIC KEY BLOCK----- xsBNBG...
NULL	NULL	NULL	NULL	NULL

Figure 6.3 Check the database if the information with public_key as been stored.

Phase3: Login page, user will enter their respective username and password if found in database then it creates a security challenge to verify its private key if it matches the public key to get approved by the server.

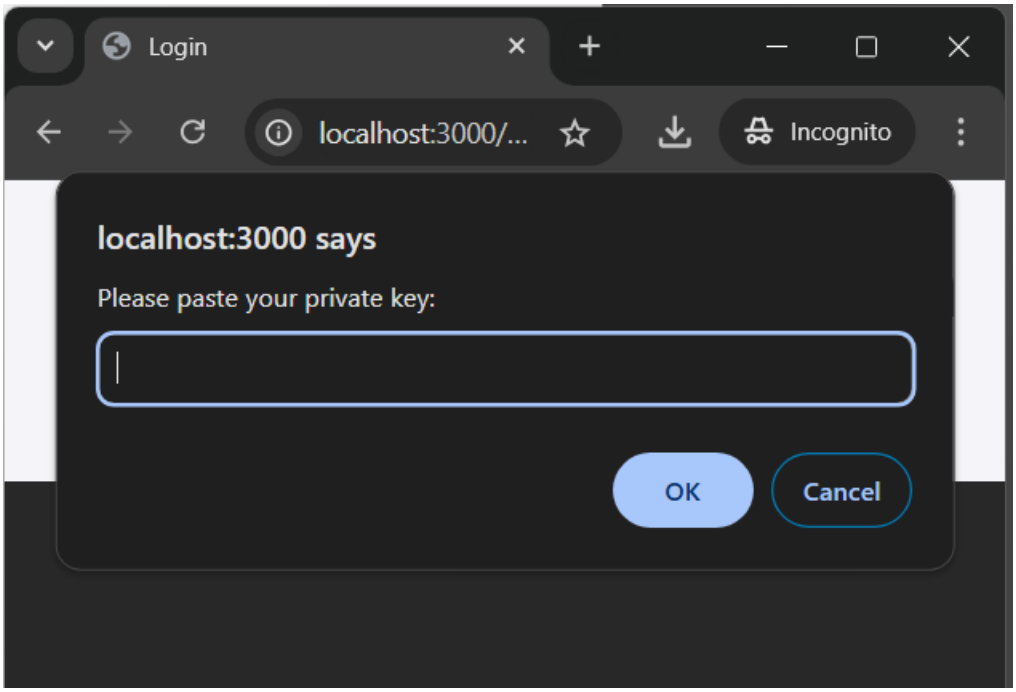


Figure 6.4 Copy paste the Private_key that was downloaded upon successful registration.

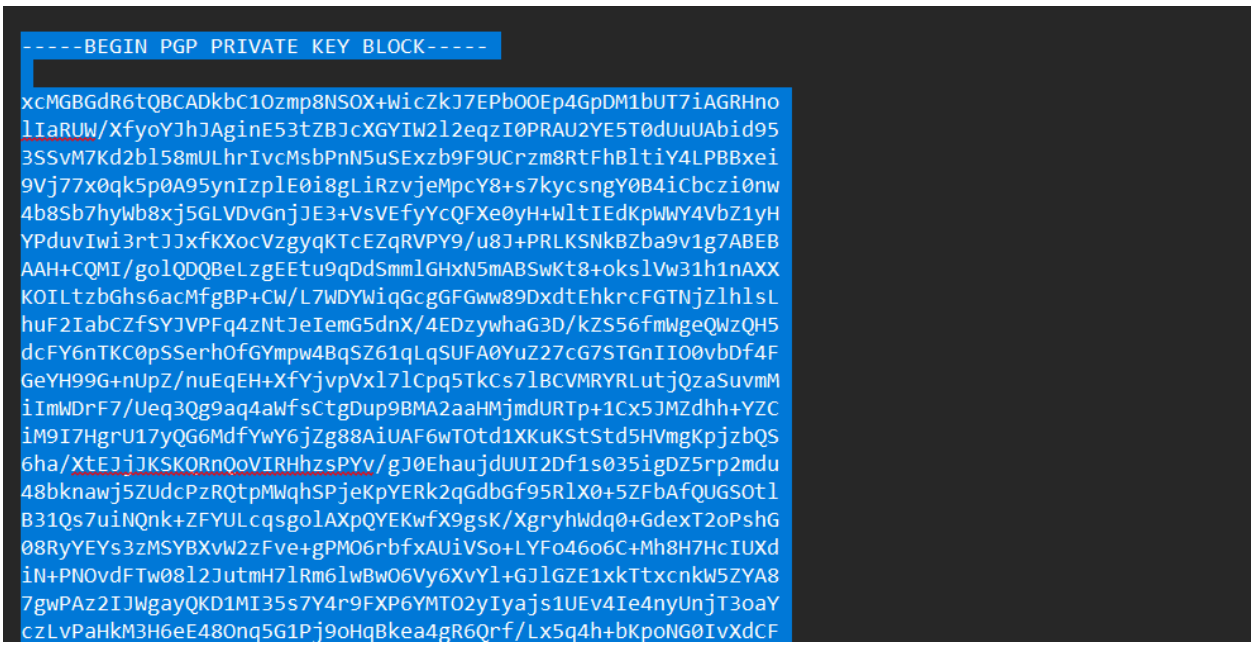


Figure 6.5 private_key

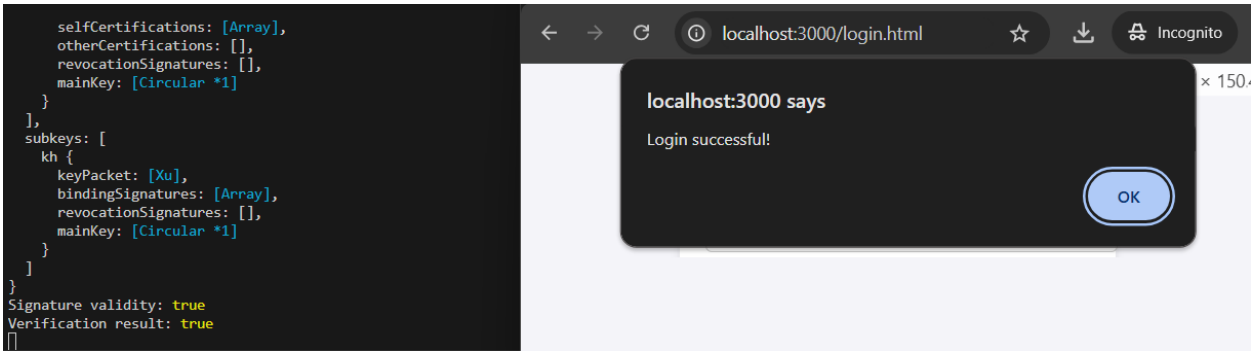


Figure 6.6 Once the Challenge is complete a popup will show Login Successful!

Phase4: The user dashboard upon successful cryptography challenge

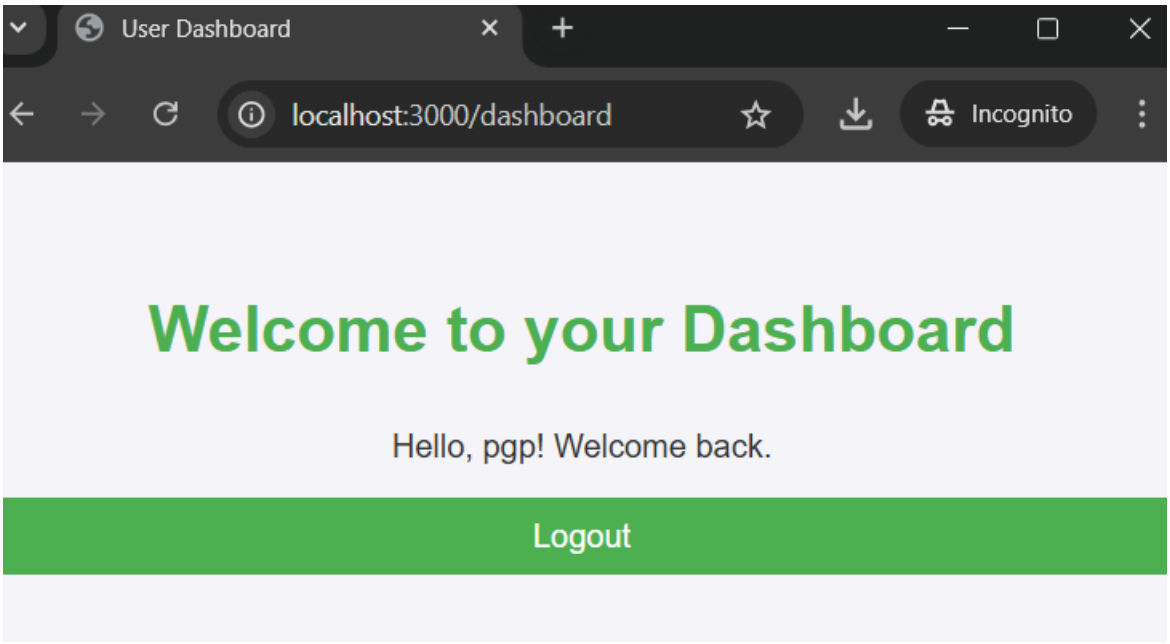


Figure 6.7 Dashboard with logout button.

References

[1] "Node.js — Run JavaScript Everywhere." Accessed: Dec. 05, 2024. [Online]. Available: https://nodejs.org/en

[2] "Visual Studio Code - Code Editing. Redefined." Accessed: Dec. 05, 2024. [Online]. Available: https://code.visualstudio.com/

[3] "MySQL :: MySQL Workbench." Accessed: Dec. 05, 2024. [Online]. Available: https://www.mysql.com/products/workbench/

[4] "openpgp - npm." Accessed: Dec. 05, 2024. [Online]. Available: https://www.npmjs.com/package/openpgp