BENCHMARKING PRE/POST-QUANTUM CRYPTOGRAPHY

USER MANUAL

BY

Cavan Phelan

C00249198

17 April 2023

Contents

1. Required Files

To ensure everything works, please make sure you have the necessary files to proceed without issue.

- Bike.java
- Dilitium.java
- Falcon.java
- Kyber.java
- Picnic.java
- Rainbow.java
- SphincsPlus.java
- AES_CTR.java
- RSA.java
- Sha3.java
- SHA256_ECDSA.java
- TwoFish.java
- Pom.xml

It is important that you have Java version 20 installed on your device. This has not been tested on other versions and issues on other versions may be expected.
It is extremely important that you have the correct pom.xml file provided as it contains all the dependencies for any plugins used on the project, otherwise, they will not be recognised. This pom.xml will be used to automatically download the dependencies when running the application, but I will still include a folder of the jar files in case they need to be added manually.

1.1. Retrieving the Files

The files are held on my GitHub, here at the steps to download the files.

- Go to my FYP repository: https://github.com/CavanP11/FYP
- On the right of the page, click the green '<> Code' button.
- Once the dropdown appears, selected 'Download ZIP' to retrieve the files.

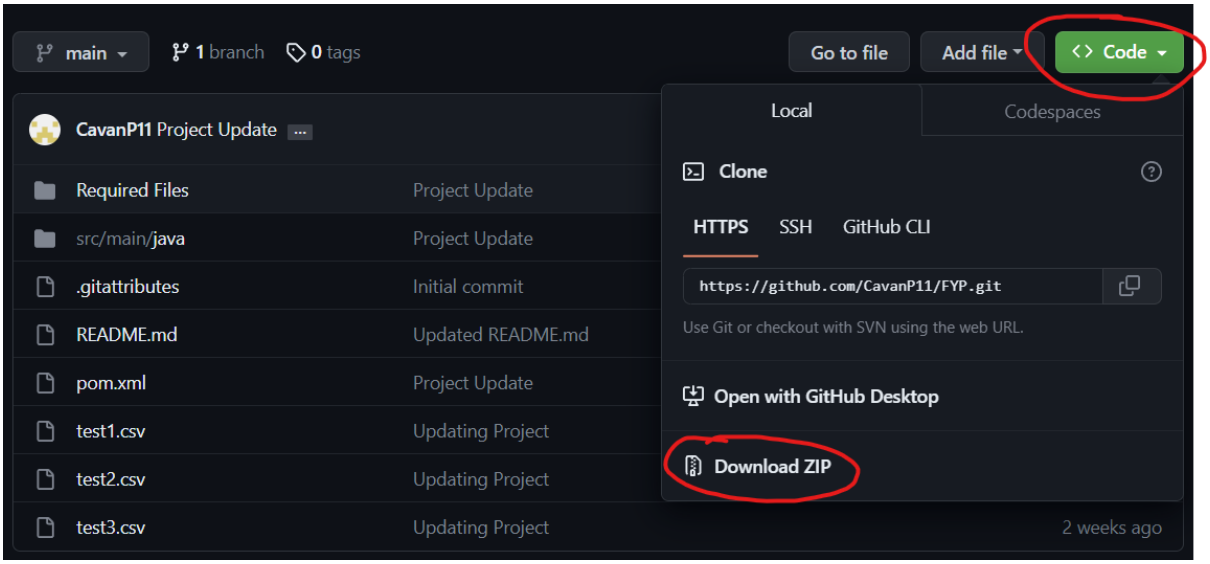

Figure 1: GitHub Files

2. Installation

This application has only been tested through Windows 10 and 11, using the IntelliJ Java IDE. Use other operating systems and IDEs at your own risk. This installation guide was done on a fresh Windows 10 VM installation.

2.1. Check Java Version

- Open your local terminal Ctrl + R on Windows and enter 'cmd'.
- Type in the following command: javac -version
- If you receive an error or a different version appears, please continue to download Java version 20.

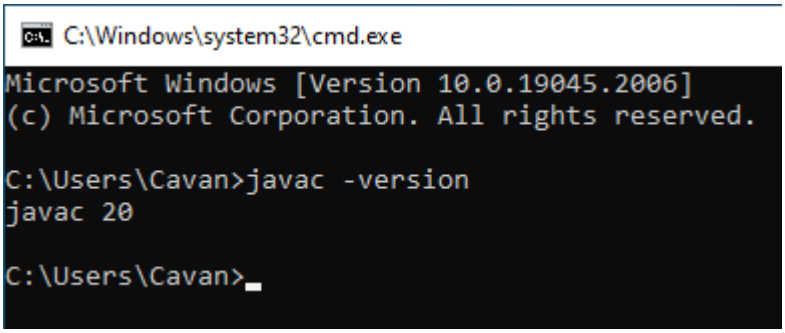

Figure 3: JDK Version

- Go to the Oracle website to download the Java 20 installer: https://www.oracle.com/ie/java/technologies/downloads/#jdk20-windows.
- Once downloaded run the installer until Java is installed.
- Run the command again to check the Java version and confirm it is version 20.

2.2. Install IntelliJ

IntelliJ has two versions, one being the free version and the other being paid. If you are a student, you can get the paid version for free, so it is up to you to choose which to use, both versions can run the application.

- Head to the IntelliJ download section and download your version of choice: https://www.jetbrains.com/idea/download/#section=windows
- Once downloaded, run the installer, and complete the installation process, you can leave everything as default.
- If you haven't already, you can register an account or log in to IntelliJ.
- Once logged in, we can create a folder on the Desktop called 'folder' and extract the zip file into this. This will be our IntelliJ workspace.

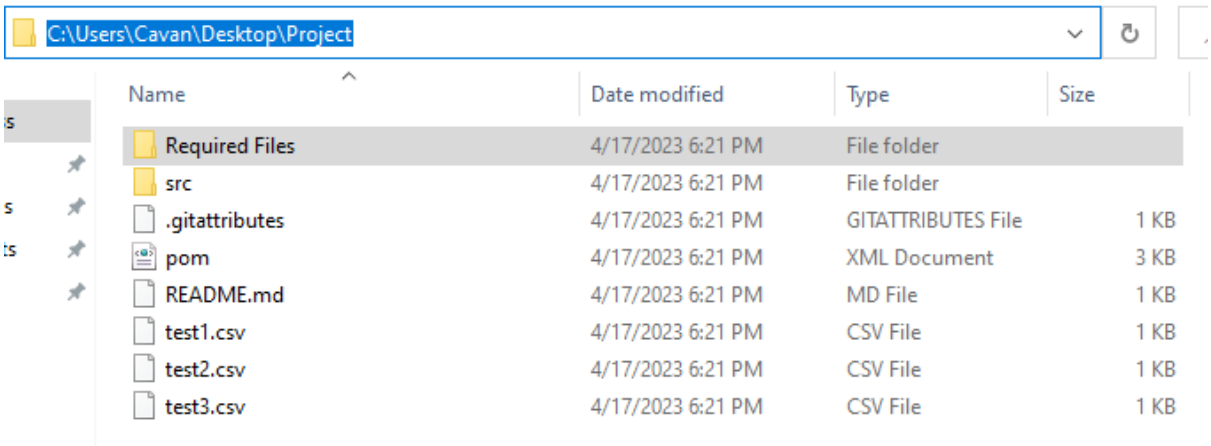

Figure 4: Folder Content

- In the IntelliJ home page, select 'Open' and open the project folder and select it. IntelliJ will give you a prompt if you want to trust and open the project, selected 'Trust Project'.

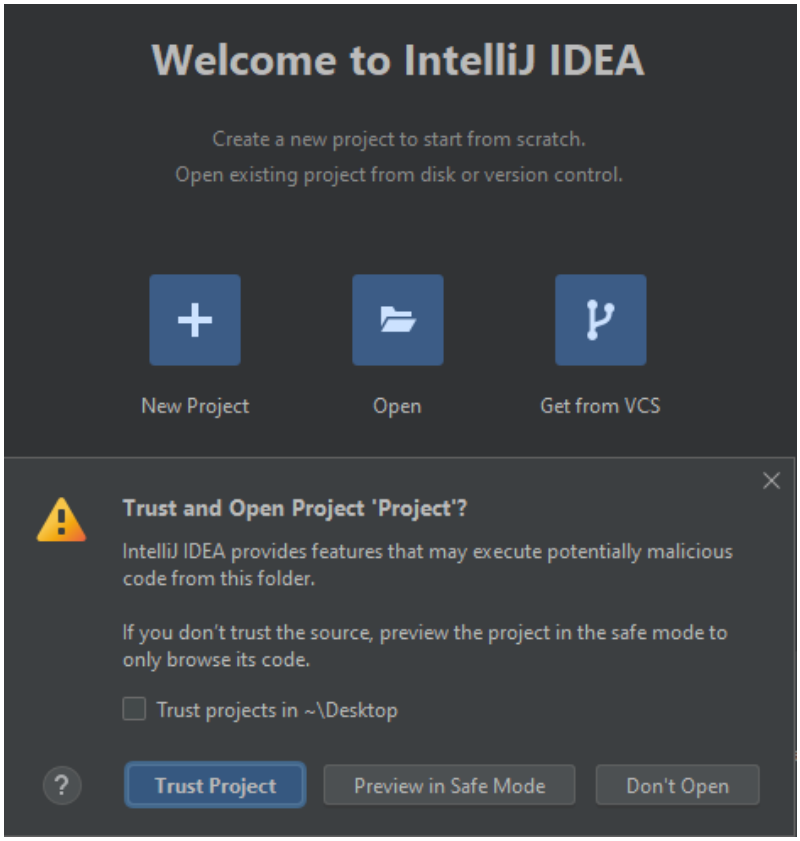

Figure 5: Trust Project

2.3. Assigning Java JDK for Project

We now need to make sure that the project is using the installed Java JDK 20. Once you've opened the project inside IntelliJ proceed with the next steps:

- In the top left of IntelliJ, navigate to 'File -> Project Structure'. This will display a new window in which we can see the project name and SDK. Make sure that the selected SDK is 20 as shown below and click 'Apply'.

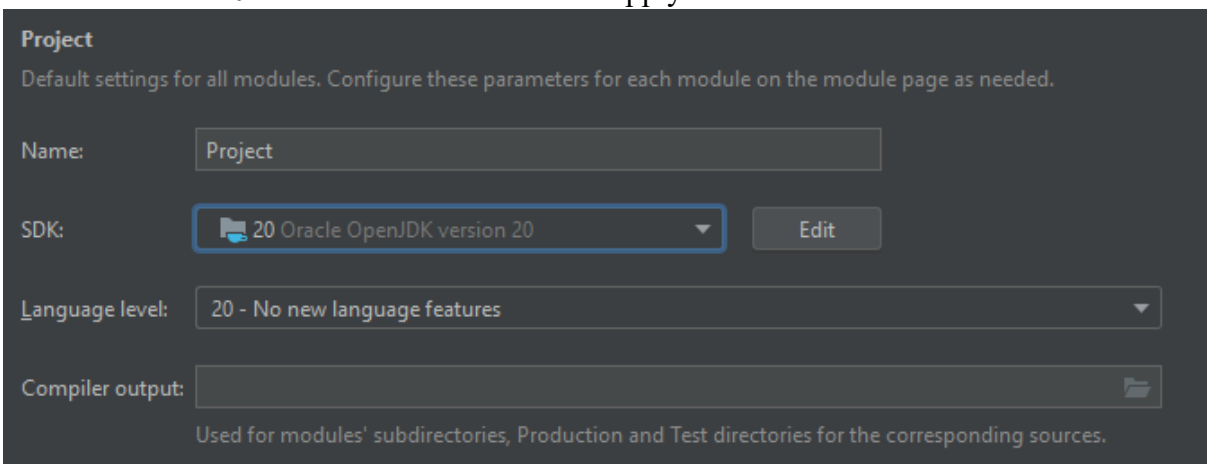

Figure 6: SDK version

2.4. Building the Project

We now need to build the project. This will install the Java 20 JDK whilst also writing the benchmark classes and dependencies. To build the project, please do the following:

- On the same navigation bar where you selected 'File', select the 'Build' option to the left and click 'Build Project'.

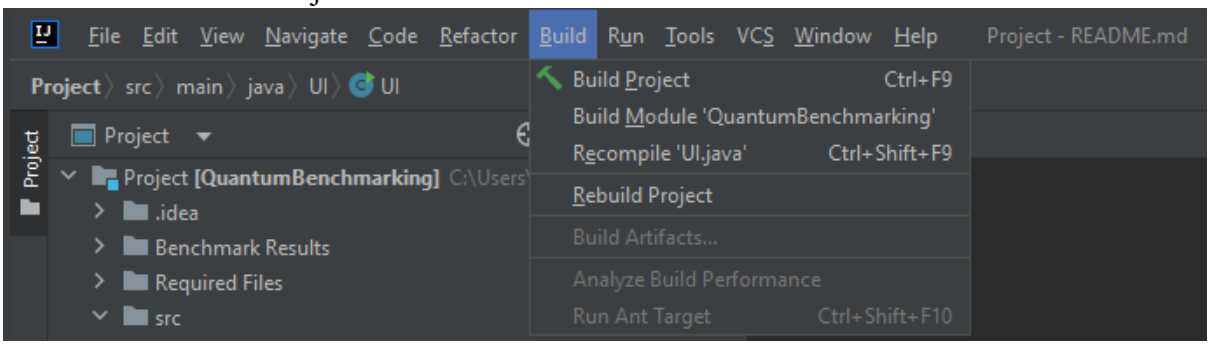

Figure 7: Building Project

- There is an installation indicator on the bottom right of IntelliJ, but you should know it's done when you have the option to run the files.

2.5. Using Maven to make sure everything is installed.

Just to be sure we have all the dependencies, we can use Maven to do its clean install, getting the information needed from the POM file I configured. To do this, please do the following:

- On the right side of IntelliJ, you will see their vertical options, 'Notifications', 'Database' and 'Maven'.

- Select the Maven option which will expand out, here we want to click the run option which is highlighted below. Here we will add ‘clean install’ and hit enter.

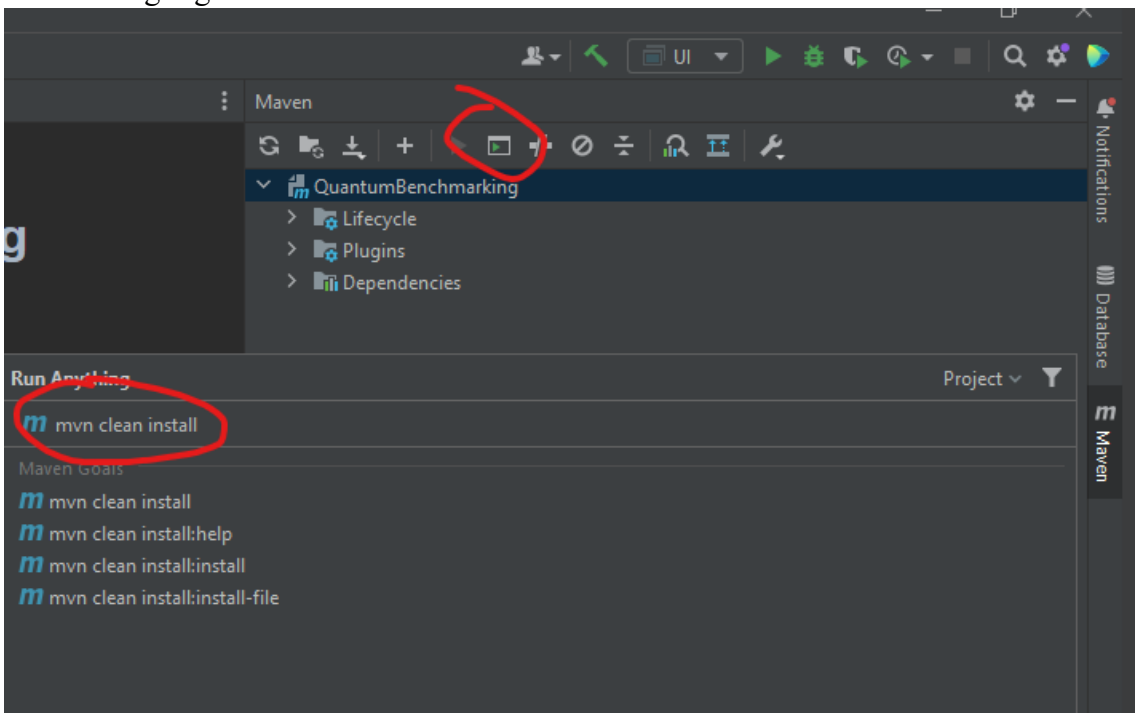

Figure 8: Maven Clean Install

2.6. Running the Application

To make sure we run the right main method, we need to find the main file,

- See the project navigation on the left to navigate to ‘src -> main -> UI ‘. Here we will see the BenchmarkUI, Graph and UI files.

- We can see a small green run button above the UI. We can right-click this and select ‘Run UI.main()’.

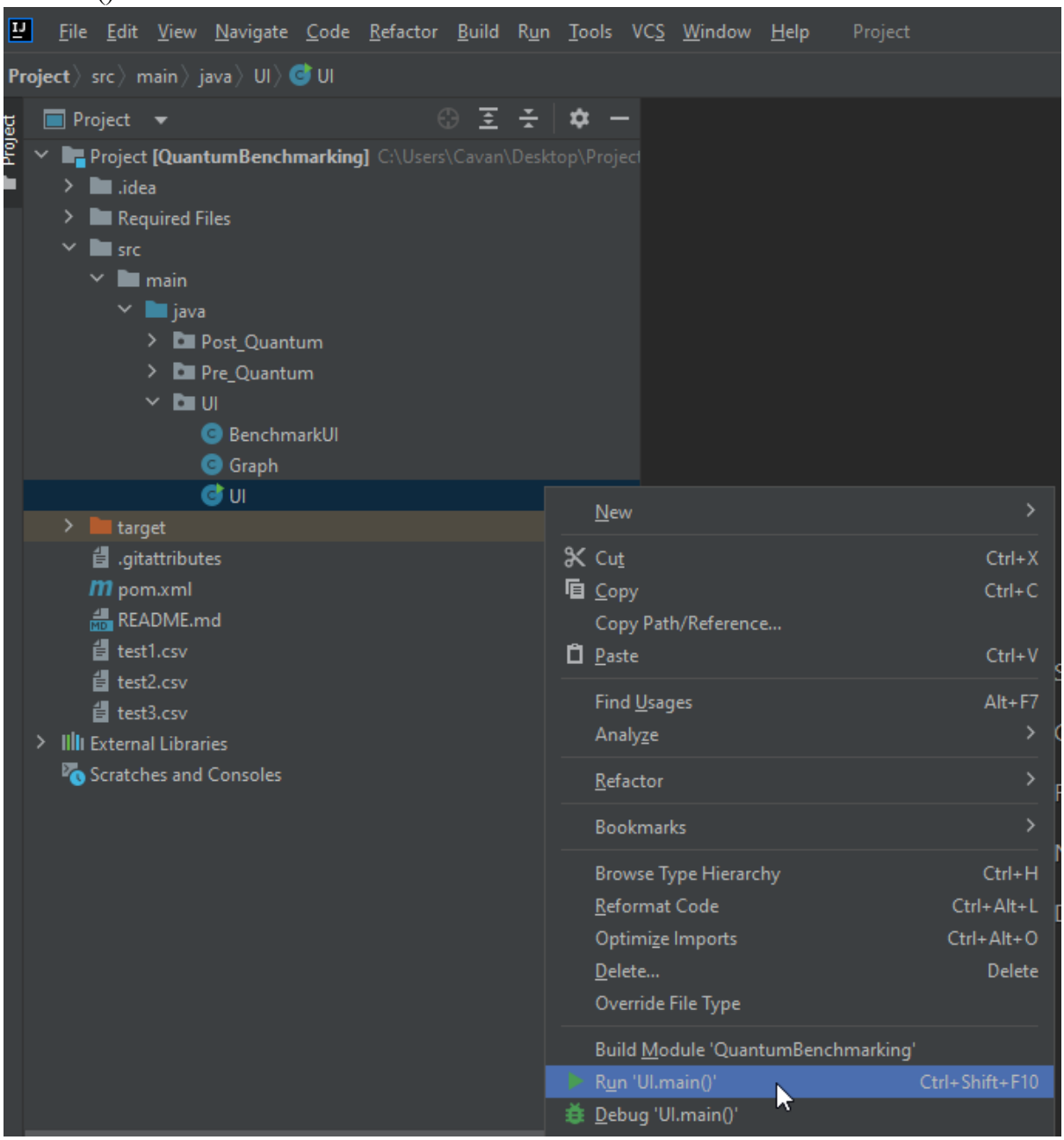

Figure 9: Running Main

- This will now run the application, and rather than having to run like that every time, we can use the start and stop buttons on the top right of IntelliJ.

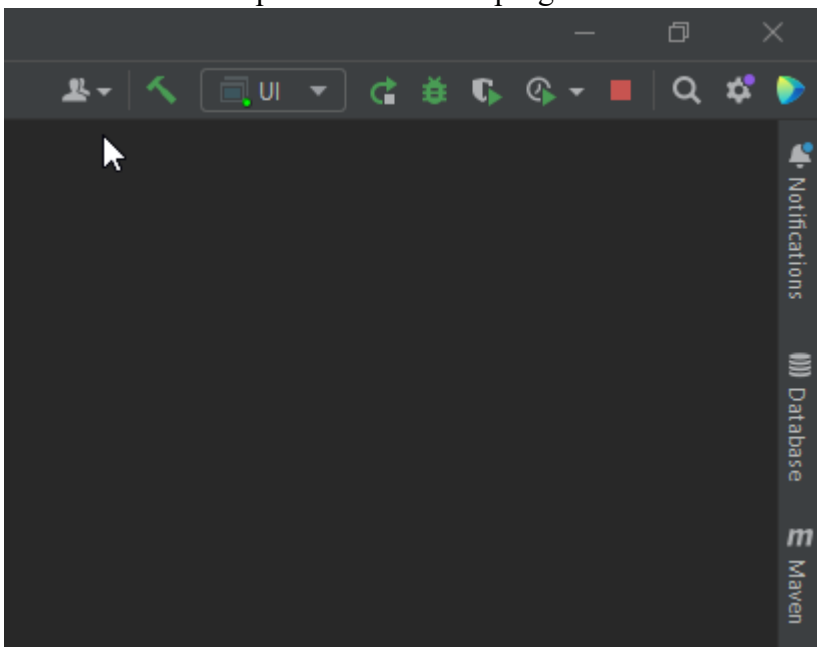

Figure 10: IntelliJ Run and Stop

3. Application Use

3.1. Home Page

Here you can select from any of the buttons, depending on your benchmarking preferences.

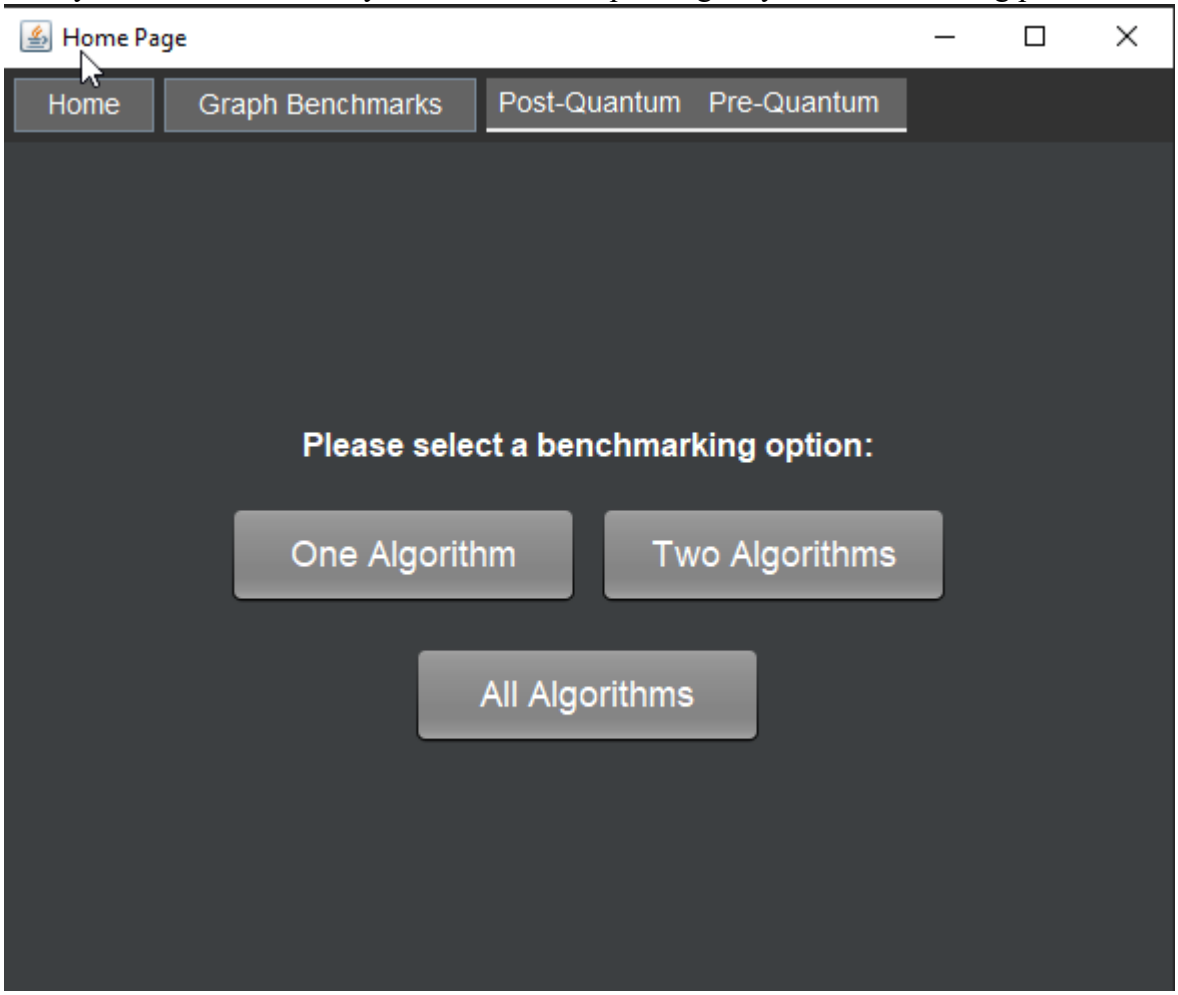

Figure 11: Home Page

All these buttons have similar functionality, you put the algorithms and select run. Except for 'All Algorithms' which you just hit run.

3.2. Choosing Algorithms

Here you will be provided with dropdown boxes containing all the algorithms you can benchmark. Once you are happy with what you want to benchmark, you can hit run to

proceed with the benchmarking

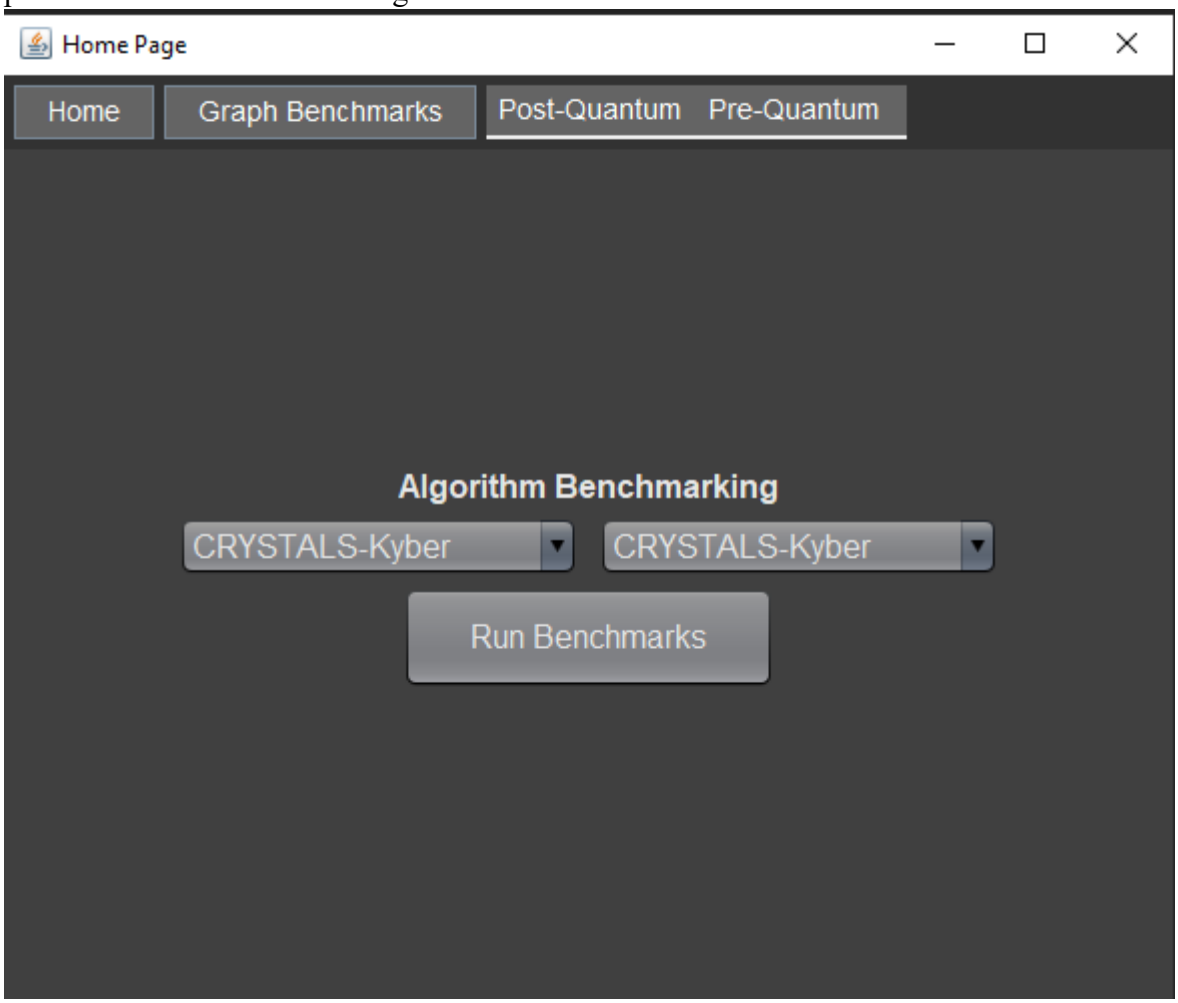

Figure 12: Algorithm Selection

3.3. Profilers

You will now be prompted to select if you want to include profilers.
****NB:** To use the ASM Profiler, IntelliJ needs to be run as an Administrator.

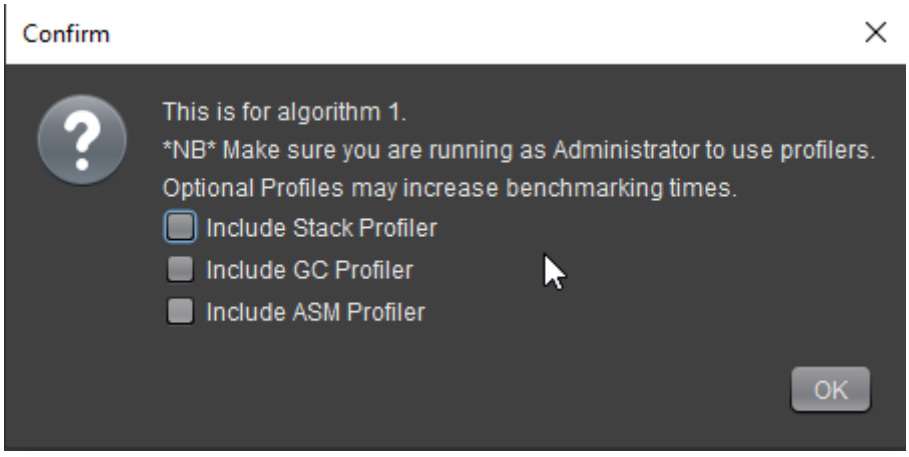

Figure 13: Profiler Options

To run IntelliJ as administrator please do the following:

- Press the Windows key and search for ‘IntelliJ’ in Windows.
- Once found, right-click on IntelliJ, and select ‘Run as Administrator’.

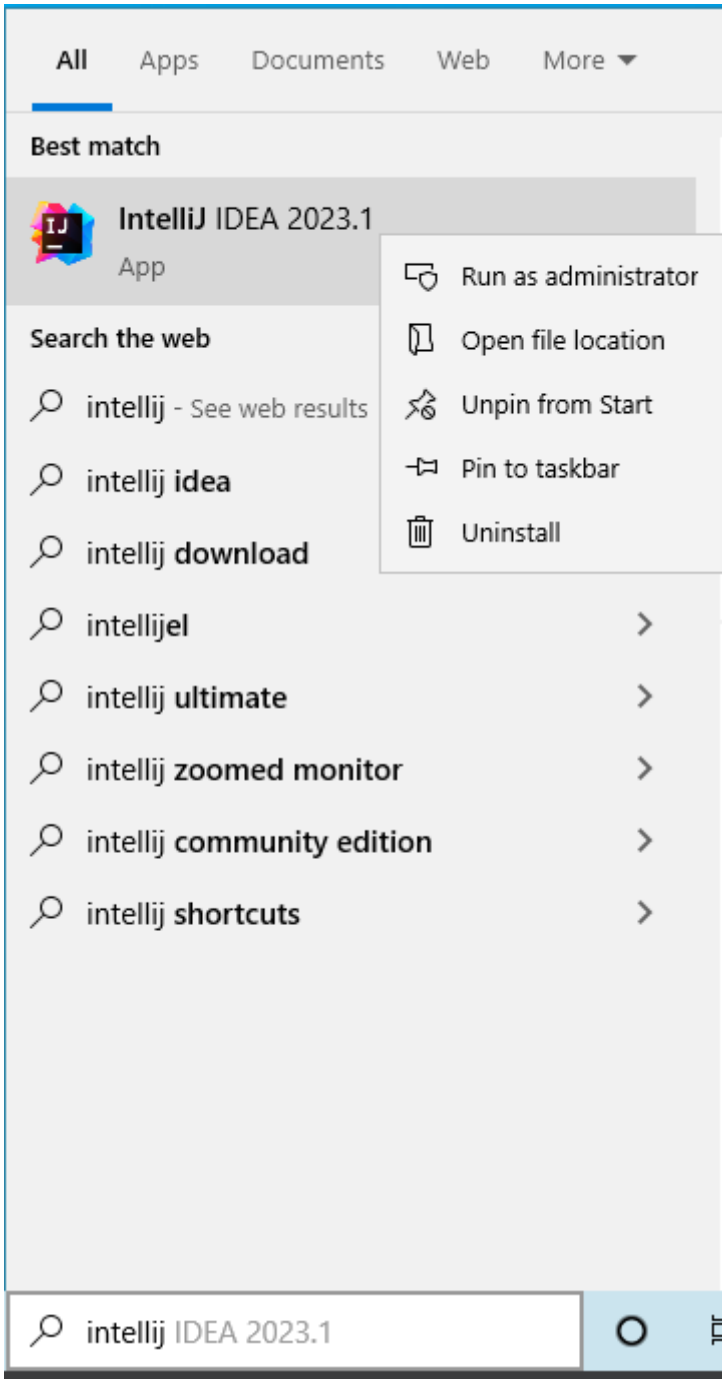

Figure 14: Running IntelliJ as Administrator.

3.4. Benchmarking

Now that you have selected the algorithms and profilers, the benchmarks will begin to run in the IntelliJ terminal, which will automatically pop up.

The benchmarks will update you on progress and estimate the time left.

```
# Benchmark: Post_Quantum.Kyber.k1024AesEncapsulatedPrivateKeyGen

# Run progress: 0.00% complete, ETA 00:06:30
# Fork: 1 of 1
# Warmup Iteration   1: 204280.921 ns/op
# Warmup Iteration   2: 171647.268 ns/op
# Warmup Iteration   3: 138733.871 ns/op
Iteration   1: 146740.798 ns/op
Iteration   2: 137756.456 ns/op
Iteration   3: 180073.758 ns/op
Iteration   4: 138228.978 ns/op
Iteration   5: 138153.651 ns/op
```

Figure 15: Benchmarking

You will know when the benchmarks are completed as it will print out the results to the terminal. You can now view these in a file in the project folder.

3.5. Viewing Benchmarks

Navigate to the projects folder, where you extracted all the files. You will see a 'Benchmark Results' folder which stores the benchmarks. Navigate to which algorithm benchmarks you wish to view. Also here, is the encoding and decoding of the algorithm's keys, signatures, and other variables it uses.

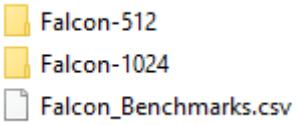

Figure 16: Benchmark File Locations

Below is an example of the Falcon Benchmark. Please note that the 'Score Error' is not calculated as this was a test run of one iteration. For the scoring error to be calculated, a minimum of three iterations per benchmark must be run.

Benchmark	Mode	Threads	Samples	Score	Score Error (99.9%)	Unit	Param: plaintextSize
Post_Quantum.Falcon.falcon1024KeyGeneration	avgt	1	1	25235820	NaN	ns/op	256
Post_Quantum.Falcon.falcon1024KeyGeneration	avgt	1	1	23631341.86	NaN	ns/op	512
Post_Quantum.Falcon.falcon1024KeyGeneration	avgt	1	1	21761787.23	NaN	ns/op	1024
Post_Quantum.Falcon.falcon1024KeyGeneration	avgt	1	1	23559788.37	NaN	ns/op	2048
Post_Quantum.Falcon.falcon1024Sign	avgt	1	1	1486132.258	NaN	ns/op	256
Post_Quantum.Falcon.falcon1024Sign	avgt	1	1	1470727.924	NaN	ns/op	512
Post_Quantum.Falcon.falcon1024Sign	avgt	1	1	1502535.135	NaN	ns/op	1024
Post_Quantum.Falcon.falcon1024Sign	avgt	1	1	1505327.055	NaN	ns/op	2048
Post_Quantum.Falcon.falcon1024Verify	avgt	1	1	74318.9129	NaN	ns/op	256
Post_Quantum.Falcon.falcon1024Verify	avgt	1	1	76440.57695	NaN	ns/op	512
Post_Quantum.Falcon.falcon1024Verify	avgt	1	1	78020.14008	NaN	ns/op	1024
Post_Quantum.Falcon.falcon1024Verify	avgt	1	1	83805.05067	NaN	ns/op	2048
Post_Quantum.Falcon.falcon512KeyGeneration	avgt	1	1	8312602.459	NaN	ns/op	256
Post_Quantum.Falcon.falcon512KeyGeneration	avgt	1	1	8100333.065	NaN	ns/op	512
Post_Quantum.Falcon.falcon512KeyGeneration	avgt	1	1	8314000.82	NaN	ns/op	1024
Post_Quantum.Falcon.falcon512KeyGeneration	avgt	1	1	8249609.016	NaN	ns/op	2048
Post_Quantum.Falcon.falcon512Sign	avgt	1	1	738585.2206	NaN	ns/op	256
Post_Quantum.Falcon.falcon512Sign	avgt	1	1	753735.0075	NaN	ns/op	512
Post_Quantum.Falcon.falcon512Sign	avgt	1	1	745875.2967	NaN	ns/op	1024
Post_Quantum.Falcon.falcon512Sign	avgt	1	1	763981.583	NaN	ns/op	2048
Post_Quantum.Falcon.falcon512Verify	avgt	1	1	40423.33293	NaN	ns/op	256
Post_Quantum.Falcon.falcon512Verify	avgt	1	1	41041.26816	NaN	ns/op	512
Post_Quantum.Falcon.falcon512Verify	avgt	1	1	45153.14779	NaN	ns/op	1024
Post_Quantum.Falcon.falcon512Verify	avgt	1	1	50479.68726	NaN	ns/op	2048

Figure 17: Benchmarking Example

3.6. Graphing Benchmarks

If you want to graph benchmarks, you can by selecting the 'Graph Benchmarks' option in the UI and then clicking 'Display Graph' like below.

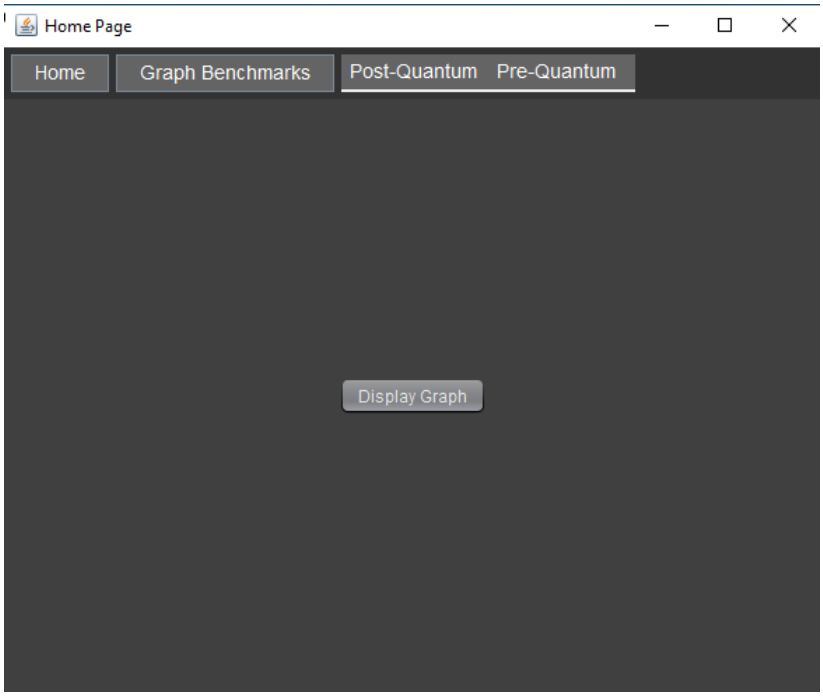

Figure 18: Graph Menu

Displaying the graph will prompt an empty canvas initially. Clicking ‘Add File’ on the bottom of this canvas will prompt you to select a .csv file.

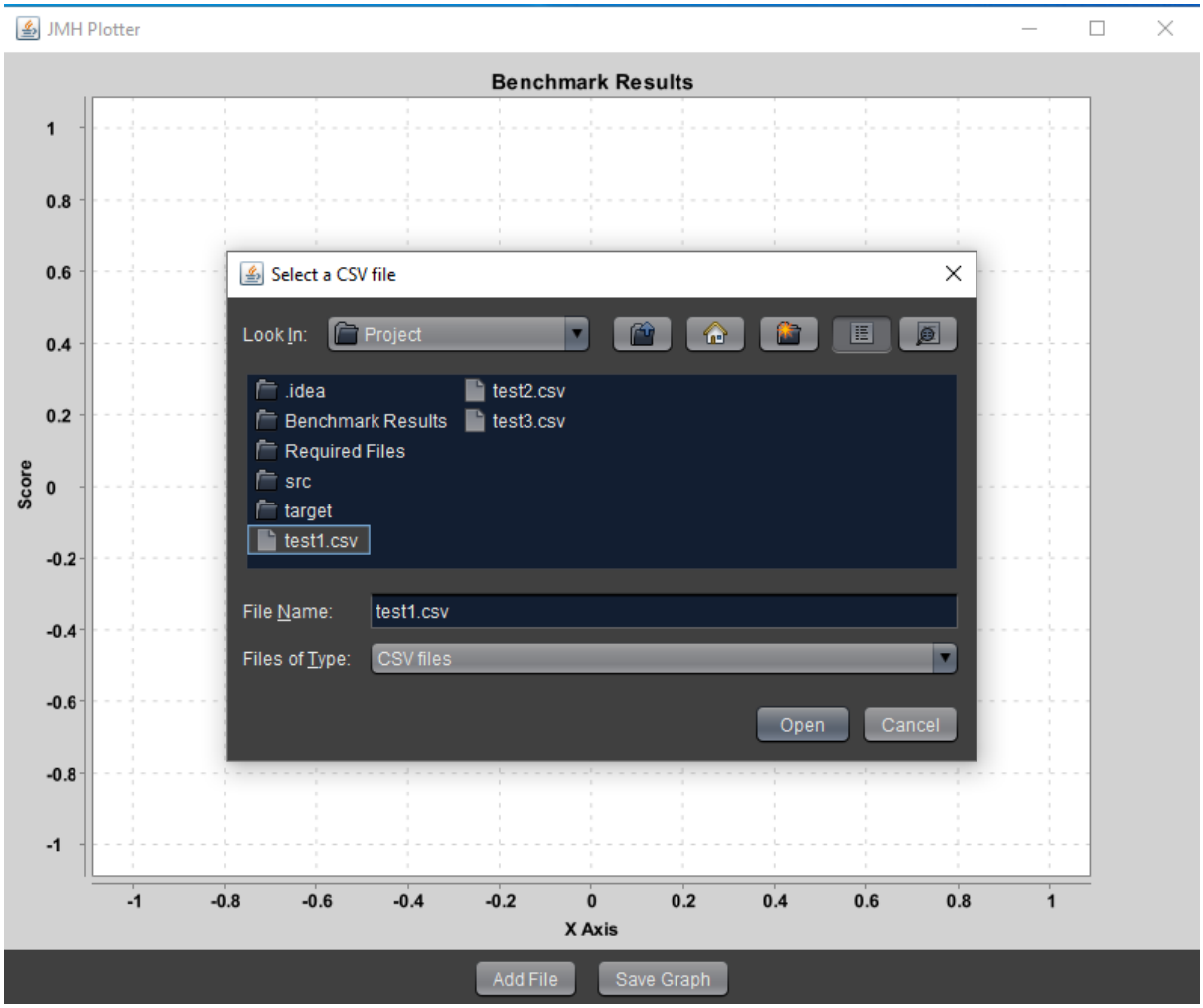

Figure 19: Selecting .csv File.

Selecting multiple graphs will plot the benchmarking scores on the graph. It's best practice to only benchmark algorithms of similar tests and types, as it wouldn't make sense to always benchmark a Zero-Knowledge Proof against a Digital Signatures, but who am I to stop you? It can still be interesting to compare.

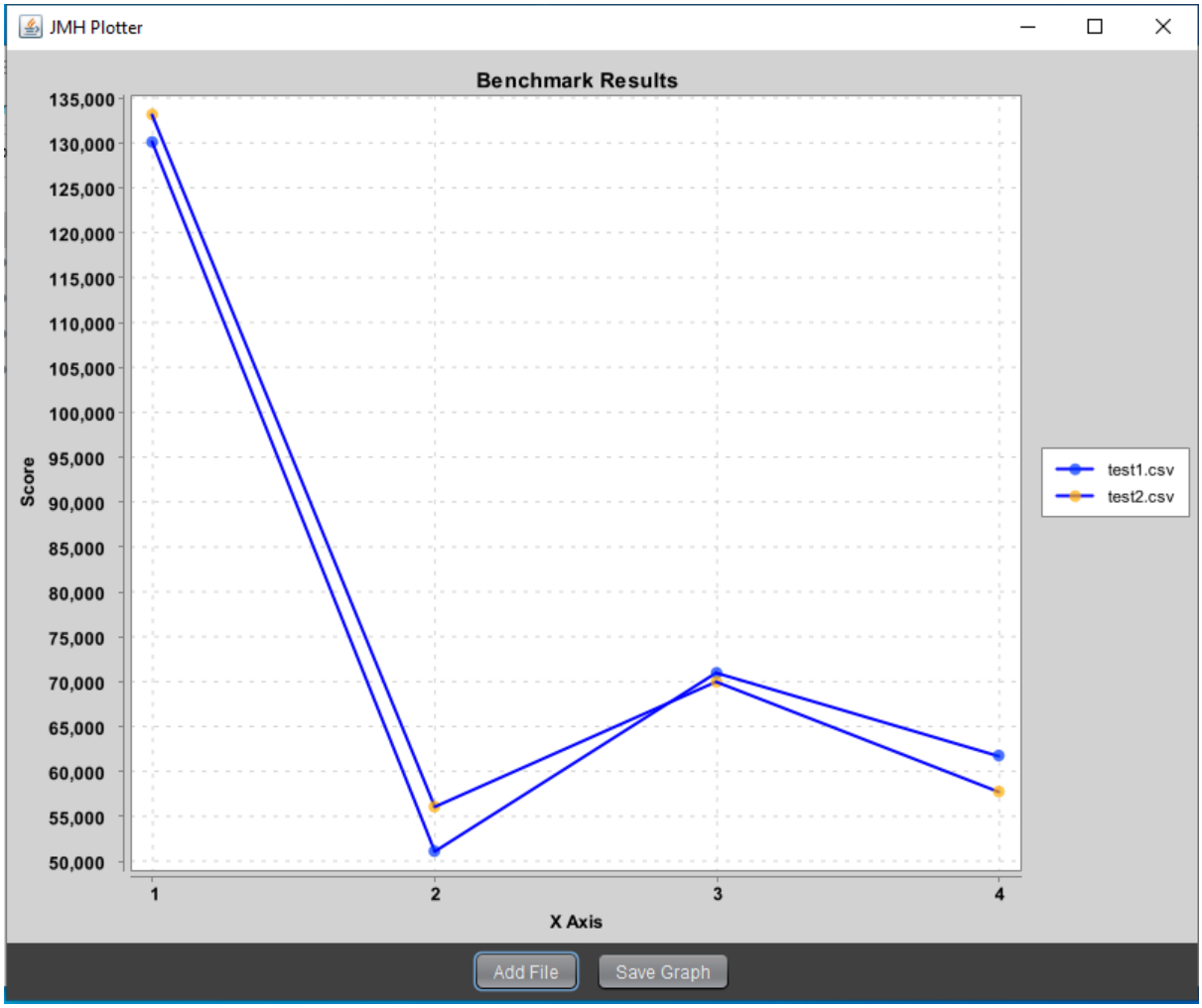

Figure 20: Graph Example

3.7. Saving the Graph

By clicking ‘Save Graph’ it will prompt you to select a location to save the file, in a similar manner as picking files to the graph. Once saved the image will be saved on the desktop to be viewed as a .png file.

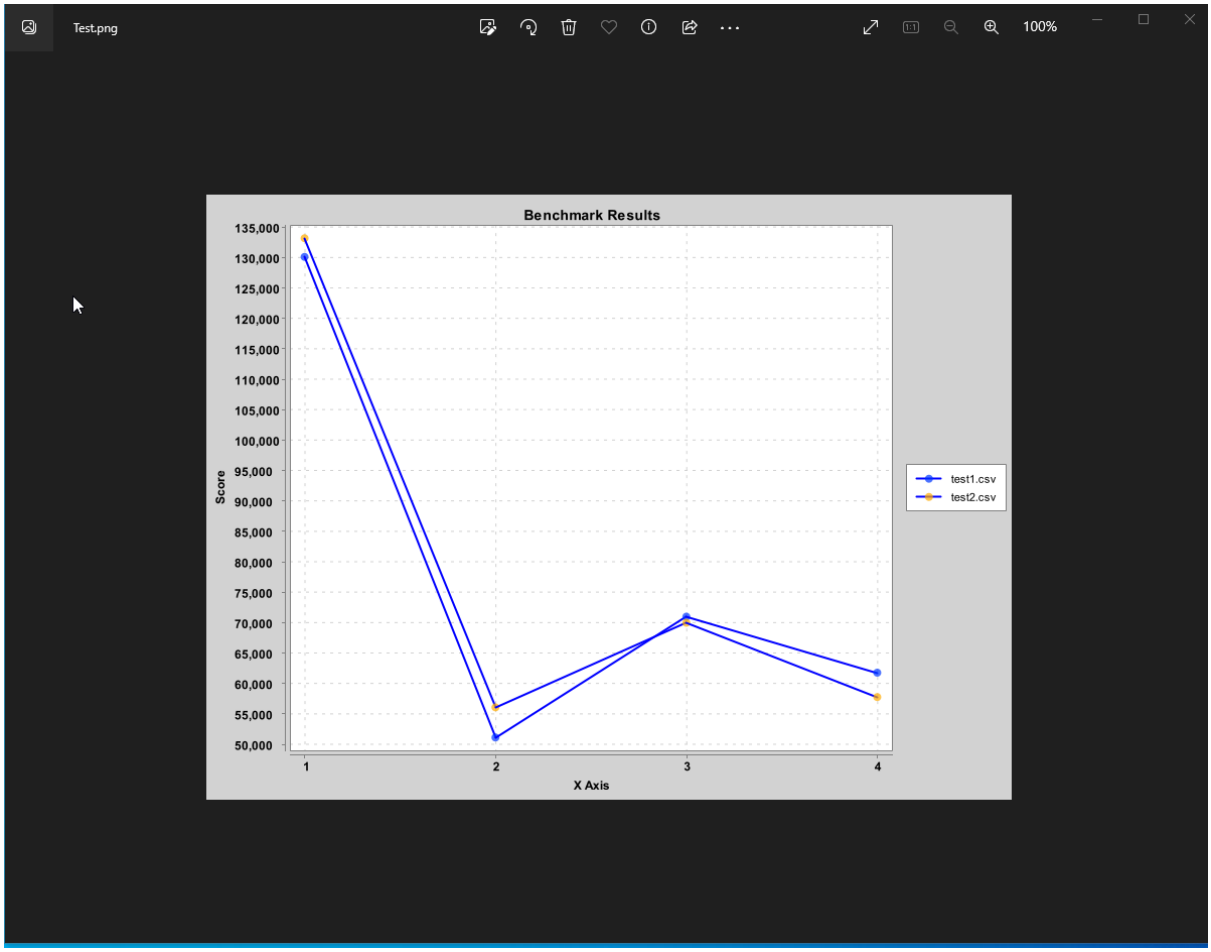

Figure 21: Graph as a .png File.

3.8. Navigating to Algorithm Websites

By hovering over the 'Post-Quantum' and Pre-Quantum' options, we see a dropdown of algorithms used to benchmark. Clicking these will either bring you to the developer's websites, or the NIST site.

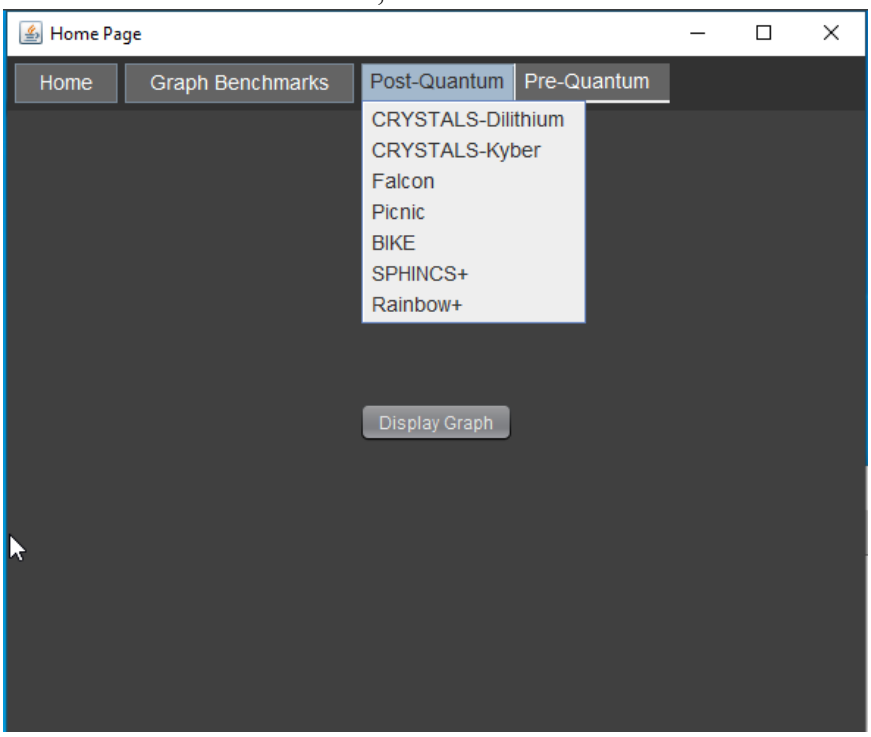

Figure 22: Algorithms

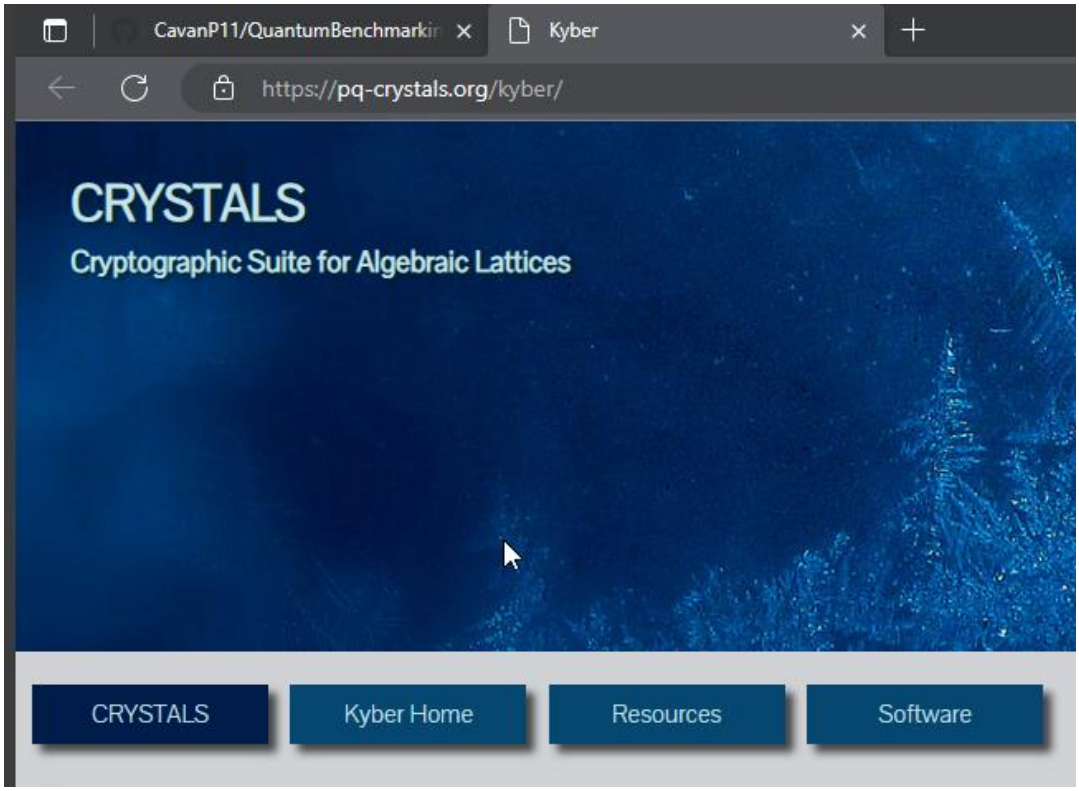

Figure 23: Developer Website

4. Troubleshooting

If for any reason you are having issues running the application, we can try manually adding the required files to the project.

- Navigate to 'File -> Project Structure' as we did before.
- Here we are going to select 'Modules' and then 'Dependencies' as shown below.

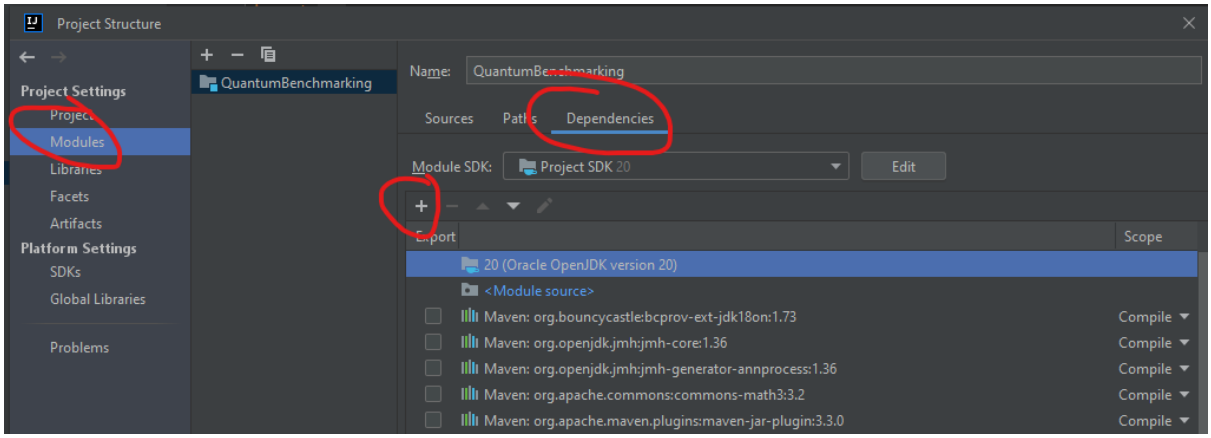

Figure 24: Dependencies

- We can click the plus symbol to now add the files. Here we will select 'JARs or Directories'.

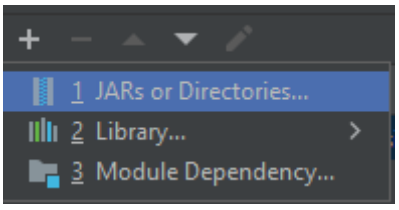

Figure 25: JARs or Directories

- The required files should be in the project workspace created at the start under ‘Required Files’.

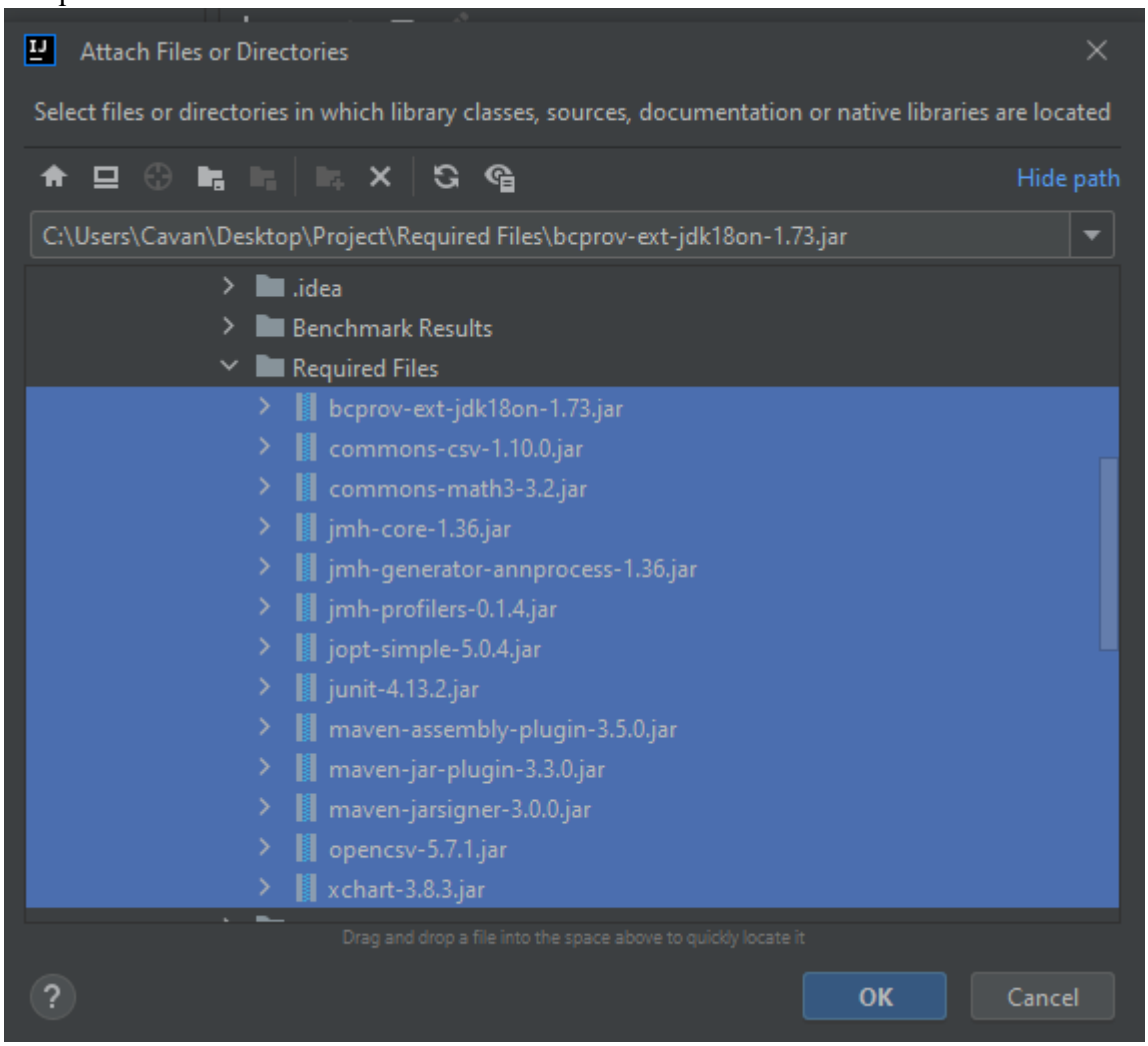

Figure 26: JAR files

- Here we can hold the SHIFT button and click the top JAR file, and then the button JAR file to select all the files. Now hit ‘OK’ and then ‘Apply’.
- Repeat the steps mentioned before when running the Maven command ‘mvn clean install’ to ensure these new dependencies are installed.