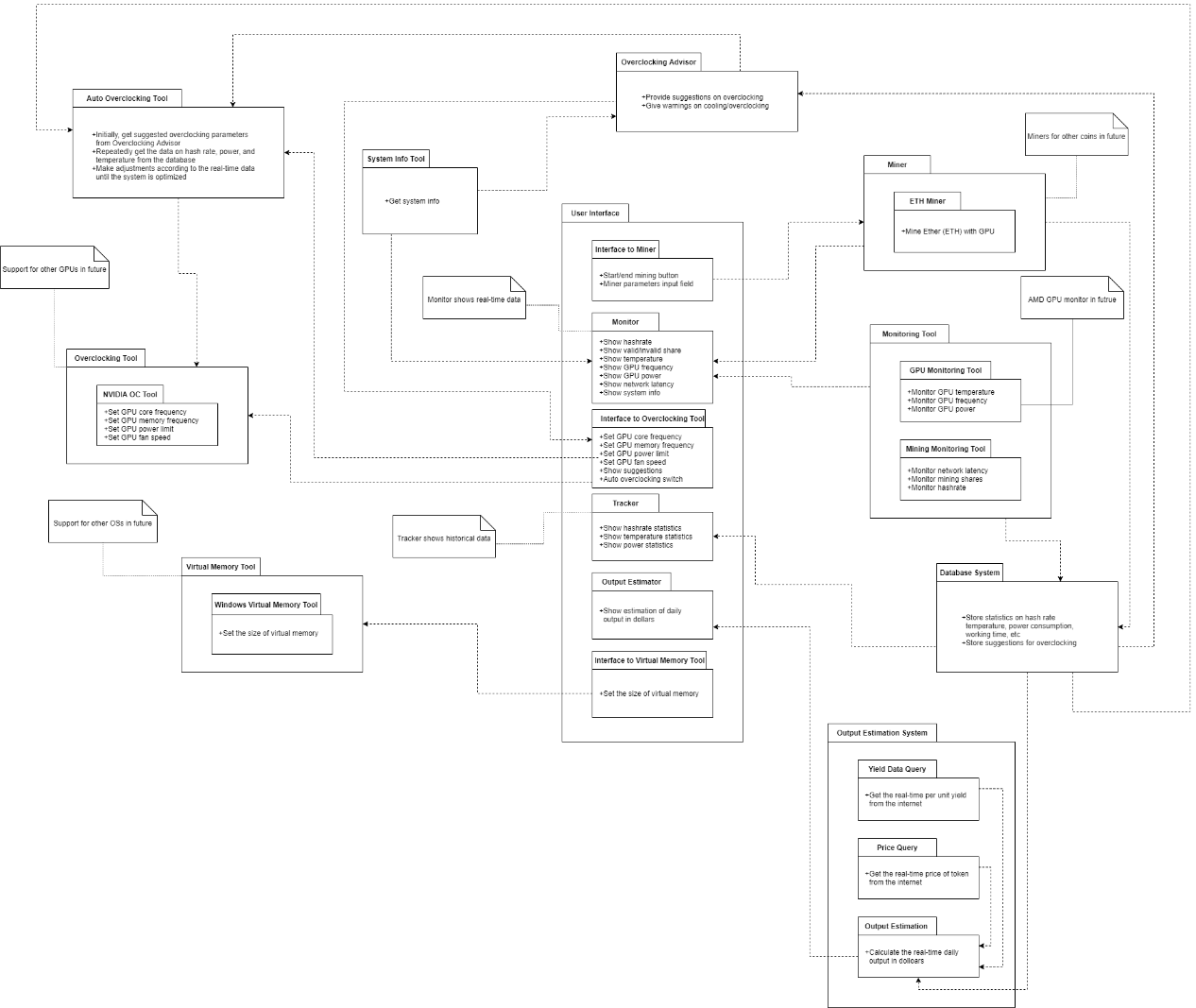
Structure of software:

Conceptual View:



Refer to Conceptual View, the main part for user to use is the User Interface, which is implemented as class “MainWindow” in the program code. While in the MainWindow, it provides connections between UI’s component and its corresponding back-end functions.

Introduction for relations between UI’s component:

Interface to Miner: It calls the interface to ETH Miner for mining.

Monitor: Monitor is used to dynamically present data and status of computer hardware and mining info. It calls Monitoring Tool to get data. Monitoring Tool is an encapsulated class to access hardware information.

Interface to Overclocking Tool: It calls overclocking objects which include overclocking advisor, Overclocking Tool and Auto Overclocking Tool to achieve the management of overclocking.

Tracker: It connects to the database system to get time series data, and do the statistic analysis. Besides, it provides hash rate of mining.

Output Estimator: It call method from Output Estimator System to show the estimation of daily output.

Interface to Virtual Memory: It call the virtual Memory Tool which is a class that contains the method that calling windows command to set virtual memory.

Design Quality in Availability:

In comprehensibility, firstly, the program shows all the related information that is related to Computer’s info, GPU’s info and Mining’s info in the User Interface in a visualize way. This is helpful for user to make sure their devices for mining are in a safe environment. Secondly, by connecting to the internet, the program will show mining hash rate and predict outcome through UI, which is helpful for users to catch the information they need. All these data are represented in user-friendly way.



Figure 1: visualized data& information

In operability, most of complex functions are encapsulation and can be called by a button. For user, it is portable to achieve most of the functions they need. For some functions which need special input, program has already given user tips to input such as sliders for numeric input (by which the range of numeric input is controlled to be valid); for some input that is limited in a set, a drop-down box is provided to user for selecting. Therefore, a user can easily learn how to use program safely without breaking down the program. Besides, compared to using MySQL, SQLite doesn’t need user to input their database ID and password, which is easy for operating and user friendly.

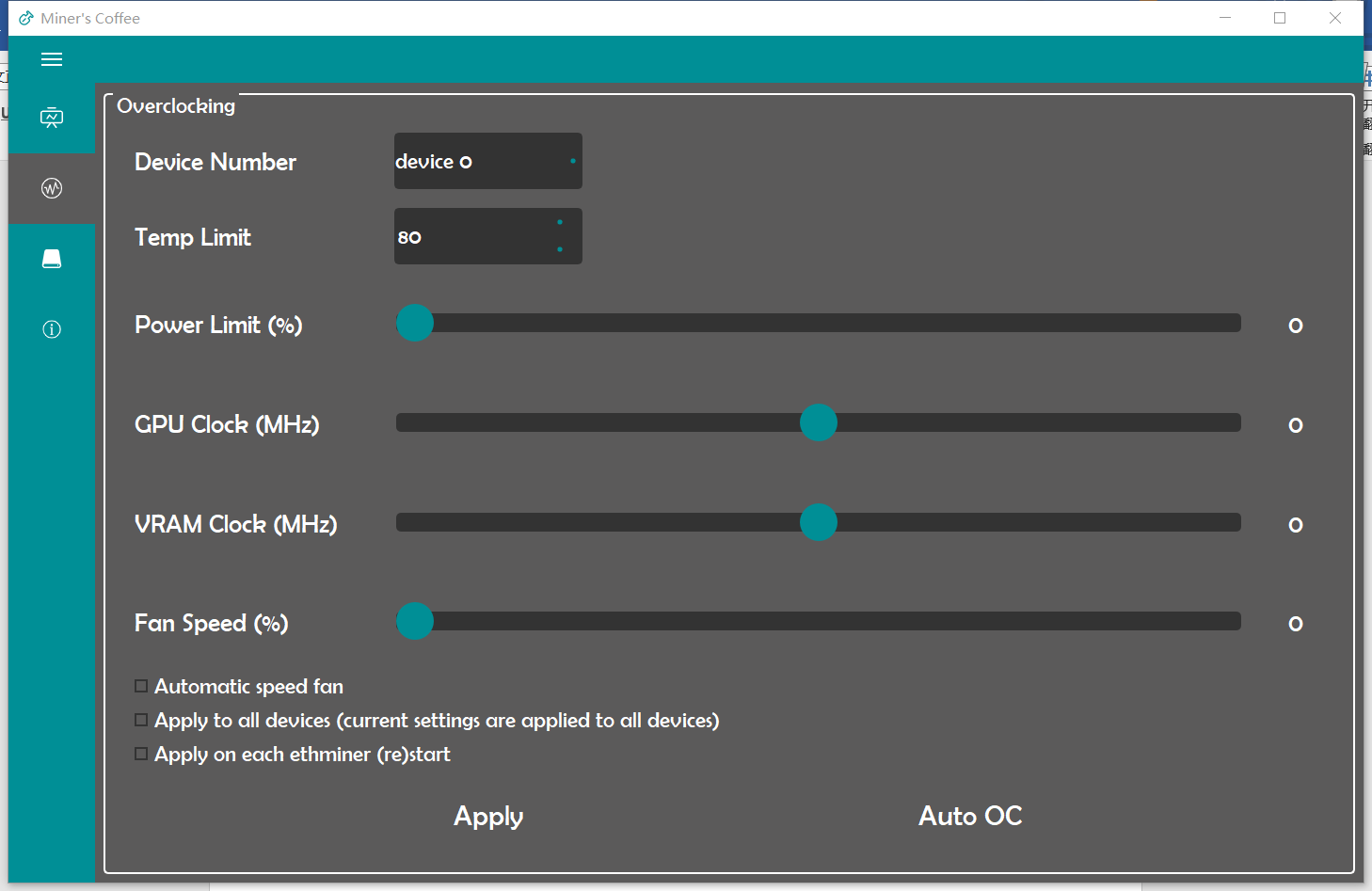


Figure 2: User-friendly Input tool

Design Quality in Maintainability:

In analyzability, most of the unit functions from back-end of the program will return its status information, by obtaining the status. By this detail feedback, it is helpful to analyze the backward or bug of the program so that programmers can find out the solutions for which is not caused by syntax error but logic or environment error. Besides, the status information and debug information provide convenience for further updating functions of the program.

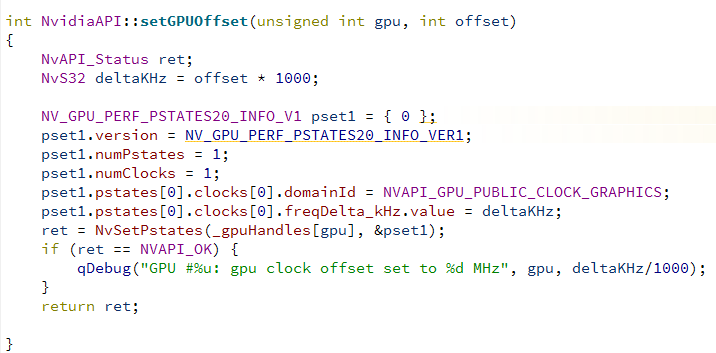


Figure 3: Detail feedback information

In testability, a testing program of Miner’s coffee is built by framework QTestLib provided by QT, it gives the testing results which show the usability of the functions, components and the full system. By applying this program, the software can be conveniently tested to confirm its usability.

Design Quality in Extensibility:

In changeability, the unit functions in the classes are clear, which is convenient for changing in the future. The relations between functions and functions or between class and class are concise. The class are independent to make sure it instantiates clearly. The class for accessing Nvidia do not have any relation with the other class such as accessing Database. Similar to the unit function, these concise structures reduce the workload when programmers try to add/ delete or adjusting functions.

Design Quality in Understandability:

In learnability, the program’s menu has summarized the functions of each page. User can easily lean the functions of each page. Each page has it clear function, which makes sure the information will be in mess. For example, the monitoring page will only show the visualized data that related to the mining info and GPU’s info. The overclocking page will only contain the information for adjusting overclocking. It makes sure the variety of information will not confound user. User can easily learn the functions of program page by page.

Design Quality in Re-usability:

In re-usability, most of the code has been encapsulated in class. Most of the accessing data query are encapsulated as a function. For example, a query from the database; a method to get GPU temperature data from Nvidia thermal sensor; a method to set overclocking of GPU or Memory; a structure in UI to show visualized data; a self-made interface to get current mining data from the internet, etc. These functions are always reuse as they encapsulated the bottom layer’s code so that it can be conveniently to use. When calling needed functions, programmer just need to instantiate a class and call its corresponding method.

Design Quality in Performance:

Considering resource utilization. In Miner’s coffee software design, some objects are dynamically created when there is need to use it, and delete it when it is no need. For example, Nvidia API is dynamically created when user calling the functions of setting overclocking. After setting, object of calling Nvidia API will be delete for better resource utilization. Besides, during monitoring the data, a thread is created, for more effective monitoring, the re-using of object NVML which get information from computer save a lot of memory. Besides, by creating multi threads, the program can achieve accessing database, monitoring computer hardware info, and mining at the same time, which enhance the effectiveness of program.