Automatski's Quantum Computer API is partially compatible with Qiskit. So we started by using the ECDLP example mentioned at https://p51lee.github.io/quantum-computing/qiskit-ecdlp/ and https://colab.research.google.com/drive/1w5DFKPIMQemzDK3x1xzq-8omYj7hMcjI?usp=sharing#scrollTo=UVej-tvnc1EE

But the quantum circuit in the example contained mid circuit measurements. And Automatski' QC doesn't support that, it only supports measurements at the end of the quantum circuit.

```
if k < 2 * NUM_BITS + 1:
    with circuit.if_test((creg_qft[k], 1)):
        circuit.x(qreg_x[0])
```

So we had to reengineer the program to not use any dynamic circuits. We then changed the code for transpilation, call to the backend and the grabbed the results returned from the backend. This enabled us to run the blog example on Automatski' quantum computers.

We went into the curves.py program and found that the parameters EC_A has to be 0 and EC_B has to be 7. Because that's what the curves.py generator used to generate the curves.

Then we set the parameters for the 7 bit curve in the code as below and ran it.

```
'''

EC_MODULUS = 67

EC_A = 0 #fixed for the QDayPrize

EC_B = 7 #fixed for the QDayPrize

EC_ORDER = 79


NUM_BITS = math.ceil(math.log2(EC_MODULUS))


N_COUNT = 2 * NUM_BITS + 2


point_p = (48, 60)  #Generator POINT

private_key = 56

'''
```

But it gave errors that the number of ancilla qubits are less. We set the number of ancilla qubits to a large random value but got an error that there were excess ancilla qubits in the circuit. So we built a workaround routine called 'find_minimum_ancilla' to find the minimum number of ancilla qubits that worked, which we then set and generate the main quantum circuit and run the rest of the program. This routine starts from #ancilla = 3 * NUM_BITS, and creates the circuit, if it gets an error it increases #ancilla by one. If it succeeds it stops and the rest of the main program starts.

There is also and int to float conversion error in the qiskit_ecdlp\impl\util\
modular_inversion_coefficients.py

Which prevented us from running large bit curves.

The qiskit-ecdlp was meant for small bit curves only. It contains multiple powerset logic inside it which causes a combinatorial explosion. This is the second reason that we couldn’t run bigger bit curves. There is no problem with the backend. The problem is in pre processing and post processing. The classical parts of the algorithm are very slow and take hours.