Some Project Ideas

- **Quantum Algorithm**
  - Implement a known quantum algorithm(VQE, QAOA, QML, QNLP, Hamiltonian Simulation, etc)
    - Implementing a quantum algorithm on your own is a great way to get your hands dirty with quantum computing. We encourage beginners to consider this option.
    - The Qiskit Textbook is a great place to start and is particularly helpful for these types of projects. It covers many algorithms and is mostly beginner friendly.
    - Qiskit also has a YouTube channel with many tutorials and lectures.
    - Nielsen and Chuang (first three chapters) is also a great resource for understanding the basics of these algorithms. It does get into more math and reasonings and is not quite necessary for the purpose of this Hackathon.
  - Taking it a step further: Benchmark the performance of an algorithm using different simulators or hardware backend
    - Many quantum computing companies/startups provide a backend that gives you access to their actual device(or simulators)! Try testing your algorithm's performance across various platforms. Some of these companies include IBM, Quantinuum, Amazon, IQM, Google.
  - Design a game that demonstrate a concept in quantum computing
    - Here's one for example!
    - And here are a few more!
- **Quantum Software-hardware Interface**
  - Qubit mapping optimizer
    - Before quantum algorithms are executed on a quantum computer, it needs to be compiled. That is, mapping the qubits from our algorithms to the physical qubits. Some algorithms favor a specific qubit topology, and how to optimally run an algorithm on a specific hardware is a really interesting problem itself. Here's a paper for reference.
  - Hardware-efficient Quantum Algorithms
    - Due to the limitations of contemporary(this is sometimes called Noisy Intermediate-Scale Quantum or NISQ era) quantum computing hardwares, many quantum algorithms(like Shor) can not be implemented with high efficiency. Recently, there has been much effort in designing quantum algorithms while considering the types of hardwares we have in the lab(surprisingly, hardware and software used to be very separated). This includes considering the limitations of the NISQ devices. Some of these ideas include algorithms run on the pulse level or algorithms that are inherently robust to noise. Consider Hardware-efficient VQE or this pulse-based VQE paper.
  - Quantum Error Correction

        - All quantum computers we have in the lab right now are subject to noise(such as decoherence, crosstalk, etc). These noises cause loss of information during computation and impede us from building a fault-tolerant quantum computer. This is where quantum error correction comes in. Quantum error correction is a prominent field in quantum computing and is theorized to achieve fault-tolerant quantum computing. The Qiskit Textbook has a few sections on quantum error correction and here's a review paper on quantum error correction. There are also many good resources online which are not listed here.
- **Quantum Hardware**
    - Hardware simulator (e.g. superconducting, semiconductor, ion trap qubit)
        - This option would be a bit more challenging. You would need to have a good physical understanding of how these quantum hardwares work. Qiskit's textbook has a few chapters on how superconducting qubits work, and this is a great review paper by Prof. Oliver's group at MIT for understanding superconducting qubits. One project idea would be building a hardware simulator that allows you to do pulse simulations. You can even implement quantum optimal control that converts logical quantum gates to microwave pulse signals. QuTip would be super helpful in this case!
    - Explore quantum control with optimal pulse design
        - As mentioned in the option above.
- **Something else!! Be creative and surprise us!!**