About likely Project topics for Operating systems:

1. Monolithic kernel vs. Microkernel *(aka: the Tanenbaum–Torvalds debate)*
2. Java-based multilayered Operating Systems
3. Firewalls and Intrusion-Detection Systems & content Filtering
4. Secured kernels
5. Multithreaded kernels
6. IPC and Synchronization
7. Multithreaded processes and deadlock problem

Example of Previous projects:

1. Caching system techniques to distribute files across the RAM of multiple nearby machines
2. Understanding and implementation of scheduling algorithm in Linux
3. Optimize power usage in Linux through CPU governor modifications
4. Mitigating radiation effects in commercial processors through operating system design
5. Scheduling optimization simulation for mobile OS
6. The multi-kernel: a new OS architecture for scalable multicore systems
7. Distributed caching file system

Also, simulation based projects are welcome. Examples are:

1. Implement a fair scheduling scheme for a system represented by a server and its clients waiting on a single queue. You need to simulate the client/server queuing system in some high-level language and show how you'd adaptively approach the problem and challenge of fair-scheduling. Compare the performance of fair scheduling with RR scheduling.

2. Study various types of fair scheduling strategy in relation to multimedia service. Explore various possible ways to address the QoS issue.

3. Disk-arm scheduling suggests varieties of algorithms which are all bundled under "elevator-type algorithm". Explore these algorithms under various disk workloads and suggest realistic variations on them depending on the relative cost of seek-time vis-à-vis the rotational delay.

4. User processes may be allowed to do some memory management themselves. They may be allowed to field their own page-fault rate and use that to adjust their own memory allocation in some application-specific situation like multimedia where they are better off with their own paging scheme. Look at the research by Steven Hand (1999) in Nemesis Operating Systems and explore that concept. https://www.usenix.org/conference/osdi-99/self-paging-nemesis-operating-system

5. Consider the issue of power management. To restart a hard-disk from a hibernating mode to spinning mode is not cheap, it is expensive. CPU should be managed to save energy. Wireless communication is also a drain for system power. Suggest ways to keep a system fully functional without wasting at the same time energy.

7. A variety of different filesystems are available. Characterize their performance under various workloads. The focus in this case is on the behavior of different filesystems subject to varieties of workloads.

8. Fetching files out of RAM is always faster than accessing a hard drive. One way to improve caching is to use peer-to-peer techniques to distribute the files across the RAM of multiple nearby machines. Develop a caching system to reflect this kind of architecture.

9. A Prefetching Web Proxy. Survey all that you can find that deals with web caching and attempts to predict or forecast user access behavior. Simulate a caching algorithm using web-server logs and traces.

10. Local vs Global page allocation policy. Explore under what conditions one might have a better edge than others. Explore various ways (on a simulated environment) how an adaptive page allocation policy could be delivered and sustained.