Distributed Systems Programming

Sample questions about theory

1. Compare client-server and peer-to-peer architectures in terms of simplicity and fault-tolerance
2. Explain why idempotent operations should be preferred when designing the interfaces of a distributed system
3. Give an example of a use case for which conditional HTTP requests are useful
4. In the RPC model, explain what a client stub is, and tell what operations it is in charge of
5. Explain the HTTP/2 push mechanism and its limitations.
6. Explain how the Lamport clock mechanism could be implemented
7. Explain how the Bully Algorithm for leader election works
8. Make an example of a distributed architecture in which running a leader election algorithm is necessary
9. Explain the concept of continuous consistency
10. What is the added value provided by websockets with respect to TCP/IP sockets?
11. Explain the meaning of MQTT QoS 1 and QoS 2. What does each QoS guarantee?
12. What is the keepalive mechanism found in some protocols? How does it work?

Sample discussion questions related to the programming assignment

1. Explain if and how HATEOAS has been implemented in your programming assignment
2. Explain if and how validation against a schema has been implemented in your programming assignment
3. Explain how you followed the main design guidelines when designing the REST API
4. Show how a programmer willing to implement a client that performs a certain operation on your REST API can obtain all the information that is necessary for its implementation. Is there anything missing in your REST API documentation?