**Functional documentation**

The aim of the project is to create a versatile solution capable of running on both Windows and Linux platforms. The project encompasses two primary components: the service and the client. The service, implemented to function on cross-platform environments, initializes and awaits incoming client connections on a designated port. Upon connection, it logs the initiation time and manages disconnections by recording the termination time and calculating the session duration. The communication between the clients and the service is fortified with SSL/TLS encryption to ensure secure data transfer.

On the client side, Python-based clients are developed to interact with the service, offering users a graphical interface with distinct buttons for connection initiation and termination. Users can select either "Connect to Windows" or "Connect to Linux" based on their system preferences, facilitating seamless cross-platform connectivity. Additionally, the client interface displays connection and disconnection times, and computes the duration of each session, providing users with comprehensive session management capabilities.

The functional sequence flow elucidates the process from the user's interaction with the client interface to the transmission of network requests via sockets, leading to server acknowledgment and subsequent session monitoring. The sequence ensures smooth communication between the client and the server, covering connection initiation, data transmission, disconnection, and confirmation processes.

The detailed functional specifications delineate the implementation approach for both the service and client applications. The service focuses on initialization, connection handling, and secure communication using SSL/TLS protocols. Conversely, the client implementation encompasses initialization, connection management, and display functionalities, ensuring a seamless user experience.

The implementation plan delineates the development milestones, starting with the service implementation, followed by client development, user interface design, secure communication integration, and testing and optimization phases. Rigorous testing strategies, including unit testing, integration testing, performance testing, and user acceptance testing, are outlined to ensure the robustness, efficiency, and usability of the solution.

Lastly, the maintenance and support section emphasizes the importance of comprehensive documentation and establishes a support plan for addressing post-deployment issues and updates.

By adhering to these comprehensive specifications, the project aims to deliver a secure, efficient, and user-friendly solution for cross-platform service management, catering to the diverse needs of Windows and Linux users.