To manage user sessions within the Stock Portfolio Management System, I will use a Redis hash with the key format "userSession:{sessionId}". Each session hash will store user-specific information such as the userID and timestamp of the last action. This will allow the system to quickly access and update session data.

For tracking recent stock quotes, another hash keyed as "stockQuote:{tickerSymbol}" will be employed. This will store the most recent price and the time of the last update, facilitating real-time updates and retrievals for stock quotes without repeatedly querying the primary database.

For user portfolio snapshots, which provide a quick overview of portfolio value and performance, I will again utilize a Redis hash. With keys patterned like "portfolioSnapshot:{userID}", each hash will store data such as portfolioID, current total value, and a text representation of performance metrics.

Pending transactions, which require FIFO processing, will be stored in a Redis list with the key "pendingTransactions:{portfolioID}". The list structure is most appropriate for queuing transactions, allowing new transactions to be appended, and old ones to be processed in the order they arrived.

To maintain a leaderboard of users with the most valuable portfolios, a Redis sorted set will be used, with the key "leaderboard". Portfolio values will be the score in the sorted set, automatically ordering the users by their portfolio's worth.

Finally, for quick access to tweets related to specific users (if it is applicable to the system), a combination of Redis lists and hashes will be implemented. Each user will have a list associated with their screen name, keyed as "userTweets:{screenName}", containing tweetIDs. A corresponding hash for each tweet, keyed as "tweet:{tweetID}", will then store the content of the tweet, such as the username, text, and creation date. This setup will provide a direct way to retrieve all tweets from a single user efficiently.