Creating a litecoin wallet with the specified requirements involves using the Litecoin RPC (Remote Procedure Call) interface and Python. Below is a basic outline of how you can achieve this. Keep in mind that handling private keys and transactions involves security considerations, so make sure to follow best practices.

1. **Install Required Libraries:** First, you need to install the `litecoin_rpc` library, which allows Python to interact with the Litecoin daemon through RPC.

   bash

   ```
   pip install python-bitcoinrpc
   ```

- **Create Configuration File:** Create a separate configuration file (e.g., `wallet_config.ini`) to store Litecoin RPC connection details, such as username, password, host, and port. Here's an example configuration file:

ini

```
      [rpc]
username = your_rpc_username
password = your_rpc_password
host = 127.0.0.1
port = 9332
```

- **Wallet Implementation:** Now, create a Python script (e.g., `litecoin_wallet.py`) with the following basic structure:

python

3.
```
import configparser
from bitcoinrpc.authproxy import AuthServiceProxy, JSONRPCException

class LitecoinWallet:
    def __init__(self, config_file_path='wallet_config.ini'):
        self.config = configparser.ConfigParser()
        self.config.read(config_file_path)

        self.rpc_connection = AuthServiceProxy(
            f"http://{self.config['rpc']['username']}:{self.config['rpc']
['password']}@{self.config['rpc']['host']}:{self.config['rpc']['port']}"
        )

    def get_balance(self):
        return self.rpc_connection.getbalance()

    def send_transaction(self, to_address, amount):
        before_balance = self.get_balance()

        try:
            txid = self.rpc_connection.sendtoaddress(to_address, amount)
            print(f"Transaction ID: {txid}")
        except JSONRPCException as e:
            print(f"Error: {e}")

        after_balance = self.get_balance()

        print(f"Before Transaction Balance: {before_balance}")
        print(f"After Transaction Balance: {after_balance}")

if __name__ == "__main__":
    wallet = LitecoinWallet()

    # Example: Send 0.1 LTC to a specific address
    to_address = "your_destination_address"
    amount = 0.1

    wallet.send_transaction(to_address, amount)
```

4. **Security Considerations:**
    - Keep the configuration file secure, and do not expose sensitive information.
    - Make sure the Litecoin daemon is running and accessible from the script's host.
    - Consider encrypting the configuration file or using environment variables for sensitive information.
5. **Usage:** You can use this wallet implementation in other Python programs by importing the `LitecoinWallet` class and creating an instance of it to automate transactions.

Remember to handle exceptions appropriately and implement additional features as needed for your specific use case. This example provides a basic foundation for creating an automated Litecoin wallet using Python and the Litecoin RPC interface.