To implement an `XRPAutomatedWallet` class in Python, you can follow these steps:

1. Install the `xrpl-py` library using pip:

```
pip install xrpl
```

2. Create a new Python file and import the following modules:

```python
from xrpl.wallet import Wallet, XRPLWallet
```

3. Define the `XRPAutomatedWallet` class:

```python
class XRPAutomatedWallet:
    def __init__(self, config_file_path):
        self.wallet = self.load_wallet_from_config(config_file_path)

    def load_wallet_from_config(self, config_file_path):
        # Load public and private keys from the configuration file
        with open(config_file_path, 'r') as file:
            lines = file.readlines()
            public_key = lines[0].strip()
            private_key = lines[1].strip()

        return Wallet(public_key, private_key)

    def display_balance(self):
        # Fetch and display the current balance
        balance = self.wallet.get_balance()
        print(f"Current balance: {balance} XRP")

    def send_transaction(self, destination_address, amount):
        # Display the balance before the transaction
        self.display_balance()

        # Build and sign the transaction
        transaction = self.wallet.send_xrp(destination_address, amount)
        signed_transaction = XRPLWallet.sign_transaction(transaction, self.wallet.private_key)

        # Submit the transaction to the XRP Ledger
        response = XRPLWallet.submit_transaction(signed_transaction)

        # Display the balance after the transaction
        self.display_balance()
        return response
```

4. Create a configuration file that contains the public and private keys for your wallet:

```
path/to/your/config/file.txt:
rHb9CJAWyB4rj91VRWn96DkukG4bwdtyTh
sEd98f3a802122ec49a1c809...
```

5. Create a new Python file and import the `XRPAutomatedWallet` class:

```python
from xrpl_py import XRPAutomatedWallet
```

6. Create an instance of the `XRPAutomatedWallet` class and pass in the path to your configuration file:

```python
wallet = XRPAutomatedWallet('path/to/your/config/file.txt')
```

7. Call the `display_balance()` method to display your current balance:

```python
wallet.display_balance()
```

8. Call the `send_transaction()` method to send XRP to another address:

```python
destination_address = 'rHb9CJAWyB4rj91VRWn96DkukG4bwdtyTh'
amount = '10' # Amount of XRP to send
response = wallet.send_transaction(destination_address, amount)
```

9. Print the transaction response:

```python
print("Transaction Response:", response)
```

This is a basic example of how to implement an `XRPAutomatedWallet` class in Python. You can extend the functionality of the class to support additional features, such as managing multiple wallets, receiving XRP, and tracking transaction history.