```
def test_sign_in_failure_incorrect_password(monkeypatch):
    """
    Test failure due to incorrect password during login.
    The function should raise RuntimeError with appropriate message.
    """
    import pytest
    from unittest.mock import MagicMock
    import signIn

    driver = MagicMock()
    driver.current_url = 'https://mint.com/'
    driver.get.return_value = None
    driver.implicitly_wait.return_value = None
    driver.find_element.return_value = MagicMock()

    monkeypatch.setattr(signIn, 'home_page', lambda d: None)
    monkeypatch.setattr(signIn, 'user_selection_page', lambda d: None)
    monkeypatch.setattr(signIn, 'handle_same_page_username_password', lambda d, e, p: None)
    monkeypatch.setattr(signIn, 'handle_login_failures', lambda d: (_ for _ in ()).throw(RuntimeError('Login to Mint failed: incorrect p

    with pytest.raises(RuntimeError) as excinfo:
        signIn.sign_in(
            email='test@example.com',
            password='wrongpassword',
            driver=driver,
            mfa_method=None
        )
    assert 'incorrect password' in str(excinfo.value)
```

---Extra Details---
Filename: signIn.py
Description: Checks that an incorrect password leads to a RuntimeError with the appropriate message. Tests robustness of login
Score: 97%
Alignment: 96%
Validation Notes: Directly exercises important error-handling branch for failed authentications.