Expense Tracker App - Test Automation

Project Overview

This project automates testing for an Android Expense Tracker application using Appium with Python. The tests are executed on BrowserStack's cloud platform using real devices.

Tools Used

- **Appium**: Open-source test automation tool for mobile applications
- **BrowserStack**: Cloud platform for testing across real devices
- **Selenium**: Underlying framework for Appium WebDriver
- **Python**: Programming language for test scripts

Test Cases Implemented

1. **Registration Test** - Verifies app launch
2. **Login Functionality Test** - Tests user authentication
3. **Logout Test** - Validates logout functionality
4. **Forgot Password Tests**:
   - Valid email scenario
   - Invalid/empty email scenario
5. **Dashboard Load Test** - Checks if dashboard loads by default
6. **Navigation Drawer Test** - Tests side menu functionality

Setup Instructions

Prerequisites

- Python 3.7+
- Appium Python client: `pip install Appium-Python-Client`
- BrowserStack account (with valid credentials)

Configuration

1. Update the following in each test file:

  - `USERNAME` - Your BrowserStack username
  - `ACCESS_KEY` - Your BrowserStack access key
  - `app` - Your app's BrowserStack upload URL

2. Device capabilities can be modified in each test file's `caps` dictionary.

Running Tests

Execute individual test files:

```bash
python test_login.py
python test_reg.py
python test_logout.py
# etc.
```

Test Reports

- Tests generate reports on BrowserStack with:
  - Pass/fail status
  - Execution videos
  - Device logs
  - Network logs

Notes

- Tests are configured for Samsung Galaxy S21 with Android 11.0
- All tests use the same app build (`bs://bd814c27814f88c6119feaa1670af2a737dbb492`)
- Debugging features (video, network logs) are enabled by default

Troubleshooting

- Check BrowserStack dashboard for detailed error logs
- Verify element IDs match your app's current implementation
- Ensure stable internet connection for BrowserStack sessions