Báo cáo Update version 1

Tóm tắt: cập nhật yêu cầu 1 - chức năng đọc usecase và chuyển thành flow, có thể đọc usecase có nhánh

Phiên bản cũ:

Trong phiên bản trước, khi đọc usecase, file usecase input phải chứa đầy đủ các bước từ đầu đến kết thúc

Ví dụ:

```
Use case Name: Login
Behavior Scenarios:
Basic flow:
1. Người dùng nhập vào [#ti][#user-name]  tên người dùng và mật khẩu [#ti][#password]
2. Người dùng yêu cầu hệ thống đăng nhập với tên người dùng và mật khẩu đã nhập [#bi][#login].
3. [#valid] [#pw] Hệ thống hiển thị trang thông báo cho người dùng biết đã đăng nhập thành công [#to]

Alternate flow 1:
1. Người dùng nhập vào tên người dùng (#ti) và mật khẩu (#ti)
2. Người dùng yêu cầu hệ thống đăng nhập với tên người dùng và mật khẩu đã nhập (#bi).
3'. [#invalid] [#pw] Hệ thống hiển thị trang thông báo cho người dùng biết đã đăng nhập thất bại do sai mật khẩu [#to]

Alternate flow 2:
1. Người dùng nhập vào tên người dùng (#ti) và mật khẩu (#ti)
2. Người dùng yêu cầu hệ thống đăng nhập với tên người dùng và mật khẩu đã nhập (#bi).
3''. [#empty] [#pw] Hệ thống hiển thị trang thông báo cho người dùng biết đã đăng nhập thất bại do chưa nhập mật khẩu [#to]
```

➔ Các event 1, 2 phải được viết lại trong các alternative flow

Version 1:

Trong phiên bản này, cô có thể đọc usecase mà không cần viết lại cụ thể các event đã có. Ví dụ:

```
Use case Name: Login
Behavior Scenarios:
Basic flow:
1. Người dùng nhập vào [#ti][#user-name]  tên người dùng và mật khẩu [#ti][#password]
2. Người dùng yêu cầu hệ thống đăng nhập với tên người dùng và mật khẩu đã nhập [#bi][#login].
3. [#valid] [#pw] Hệ thống hiển thị trang thông báo cho người dùng biết đã đăng nhập thành công [#to]

Alternate flow 1:
[1. 2., basic flow]
3'. [#invalid] [#pw] Hệ thống hiển thị trang thông báo cho người dùng biết đã đăng nhập thất bại do sai mật khẩu [#to]

Alternate flow 2:
[1. 2., basic flow]
3''. [#empty] [#pw] Hệ thống hiển thị trang thông báo cho người dùng biết đã đăng nhập thất bại do chưa nhập mật khẩu [#to]
```

Thay vì viết lại các event trong basic flow. Cô có thể thay chúng bởi 1 câu rút gọn

[event bắt đầu. event kết thúc., tên flow chứa event]

Ví dụ:

[1. 12., basic flow]

[3. 9’., alternative flow 1]

Note: Cấu trúc này có thể được tùy chỉnh trong file annotation.xlsx

Kết quả output vẫn cụ thể như ở phiên bản cũ:

output > flows_v1.txt

```
1. #ti #user-name #ti #password  -> 2. #bi #login  -> 3. #valid #pw #to
1. #ti #user-name #ti #password  -> 2. #bi #login  -> 3'. #invalid #pw #to
1. #ti #user-name #ti #password  -> 2. #bi #login  -> 3''. #empty #pw #to

```

Ví dụ khác với input:

```
Basic flow:
1. Use case này bắt đầu khi khách hàng đến quầy thanh toán cùng với giỏ hàng
2. Thu ngân lưu lại mã của các mặt hàng và số lượng [#ti][#id][#ti][#number]
3. Hệ thống hiển thị giá mặt hàng và thêm thông tin bổ sung vào dòng đơn hàng đang thực hiện [#to][#price][#to][#description]
4. Thu ngân thông báo cho hệ thống biết đã hoàn thành việc nhập đơn hàng [#bi][#endSale]
5. Hệ thống hiển thị tổng tiền đơn hàng [#to][#total]
6. Thu ngân thông báo cho khách hàng biết tổng tiền đơn hàng
7. Khách hàng nộp tiền mặt hoặc thẻ ngân hàng, số lượng nhiều hơn hoặc bằng tổng tiền hóa đơn
8. Thu ngân nhập vào số tiền nhận từ khách hàng [#ti][#amount]
9. Hệ thống hiển thị số tiền dư [#to][#balance]
10. Thu ngân đưa lại tiền dư và phiếu thu cho khách hàng
11. Khách hàng dời siêu thị với các mặt hàng đã mua và phiếu thu
12. Thu ngân thông báo cho hệ thống biết việc thanh toán đơn hàng đã hoàn thành [#ti][#endSale]
13. Hệ thống ghi nhận việc thanh toán một đơn hàng đã hoàn thành [#to][#endSale]

Alternative flow 1:
[1. 1., Basic flow]
2'. [#invalid product id] Thu ngân nhập sai mã hàng -> Hệ thống thông báo lỗi [#to][#error]
3'. Thu ngân nhập lại mã của các mặt hàng và số lượng [#ti][#id][#ti][#number]
[3. 6., basic flow]

Alternative flow 2:
[2'. 5., Alternative flow 1]
6''. Thu ngân kiểm tra lại thông tin đơn hàng
```

Output:

1. -> 2. #ti #id #ti #number -> 3. #to #price #to #description -> 4. #bi #endsale -> 5. #to #total -> 6. -> 7. -> 8. #ti #amount -> 9. #to #balance -> 10. -> 11. -> 12. #ti #endsale -> 13. #to #endsale

1. -> 2'. #to #error -> 3'. #ti #id #ti #number -> 3. #to #price #to #description -> 4. #bi #endsale -> 5. #to #total -> 6.

2'. #to #error -> 3'. #ti #id #ti #number -> 3. #to #price #to #description -> 4. #bi #endsale -> 5. #to #total -> 6''.

Cách sử dụng

Cách sử dụng vẫn không thay đổi, để chuyển từ file usecase → flow, ta sử dụng lệnh cmd:

py gen_flows_txt.py + tên file usecase input (đã được đặt trong folder input) + tên file flows output

Cập nhật mã nguồn

Flow.py

Thêm hàm equal: so sánh 2 flow name

```
def equal(self, flow_name):
    n1 = re.sub('[^a-zA-Z0-9]+', '', self.name).lower()
    n2 = re.sub('[^a-zA-Z0-9]+', '', flow_name).lower()
    return n1 == n2
```

Ví dụ: Basic flow = basic flow = basicflow

Thêm hàm: Lấy danh sách các event liên tiếp trong flow

```
def get_event_list(self, start_event, end_event):
    a = self.get_event_index(start_event)
    b = self.get_event_index(end_event)
    if (0 <= a and a <= b and b < len(self.events)):
        return self.events[a : b + 1]
    else:
        print("Error: Khong co danh sach event tu ", start_event, " den ", end_event, " trong ", self.name)
```

Monitor.py

Thêm hàm: Kiểm tra 1 dòng có phải 1 chuỗi event được viết tắt

```
def is_event_list(self, str):
    str = str.lower().strip()
    event_list_regex = self.event_list[0]
    return re.match(event_list_regex, str)
```

Ví dụ:

[1. 3., alternative flow 1] → 1 chuỗi event

[2. 6.] → Không đủ thông tin để là 1 chuỗi event

Thêm hàm: Trích xuất thông tin của chuỗi event được viết tắt

```
def extract_event_list(self, line):
    line = line.lower().strip()
    event_list_regex = re.compile(self.event_list[0])
    event_list = event_list_regex.search(line).group()
    events, flow = event_list[1:-1].split(',')
    start_event, end_event = events.split(' ')
    return start_event, end_event, flow
```

Ví dụ:

Input:

line = [1. 8., basic flow]

Output:

Start event = 1.

End event = 8.

Flow = basic flow

Services.py

Cập nhật hàm đọc usecase, thêm chức năng đọc event list (chuỗi event được viết tắt)

```
def read_usecases_txt(usecases_file_path):
    flows = []
    events_collection = EventCollection()

    lines = []
    with open(usecases_file_path, mode="r", encoding="utf-8") as reader:
        lines = reader.readlines()

    for line in lines:

        if (monitor.is_flow_name(line)):
            flows.append(Flow(line))

        elif (monitor.is_event_list(line)):
            start, end, flow_name = monitor.extract_event_list(line)

            for f in flows:
                if (f.equal(flow_name)):
                    break
            events = f.get_event_list(start, end)

            for e in events:
                flows[-1].add_event(events_collection.get_event(e.name))

        elif (monitor.is_event_name(line)):
            e = Event(monitor.extract_event_name(line))
            e.set_annotations(monitor.extract_annotations(line))

            events_collection.add_event(e)
            flows[-1].add_event(events_collection.get_event(e.name))

    return {'flows': flows, 'events': events_collection}
```

Ngoài ra, có thêm các cập nhật ở file annotation.xlsx, myconstants.py

Chủ yếu là thay đổi các biểu thức chính quy để duyệt string