Đại học Quốc Gia – Đại học Bách Khoa

**Khoa khoa học và kĩ thuật máy tính**

**🙞∙∙∙☼∙∙∙🙜**

****

báo cáo Bài tập Lớn 2

Kiểm Tra phần mềm

Nhóm L02 – 03

Giảng viên hướng dẫn: PGS.TS Bùi Hoài Thắng

|  |  |  |
| --- | --- | --- |
| **Sinh viên thực hiện** | **Mã số sinh viên** | **Điểm số** |
| Trịnh Duy Hưng | 1913652 |  |
| Hà Hải Thiên Sơn | 1811193 |  |
| Nguyễn Tiến Đăng Khoa | 1832026 |  |
|  |  |  |

Mục lục

1. Lời cảm ơn 4
2. Giới thiệu 5
3. Phân công nhiệm vụ 5
4. Đặc tả ứng dụng 6

1. Functional requirements 6

2. Non-functional requirements 6

1. Kiểm tra các yêu cầu chức năng 7

1. H 7

1. 1
2. 1
3. 1
4. 1

2.

1. 1
2. 1
3. 1
4. 1

3.

1. 1
2. 1
3. 1
4. 1

4.

1. 1
2. 1
3. 1
4. 1

5.

1. 1
2. 1
3. 1
4. 1
5. Kiểm tra các yêu cầu phi chức năng

1. 5

2. 7

3. 8

1. Tổng kết 11
2. **Lời cảm ơn**

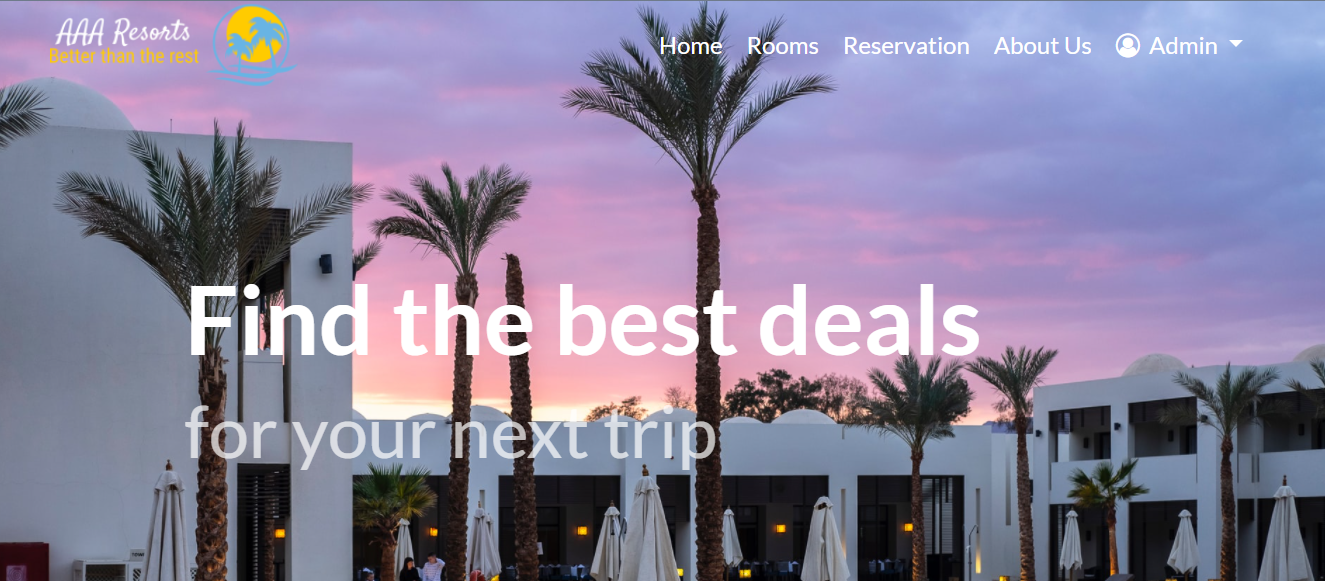
Thân chào thầy và các bạn sinh viên, đây là bài tập lớn môn Kiểm tra Phần mềm với sự hướng dẫn của thầy Phó giáo sư tiến sĩ Bùi Hoài Thắng.

Chúng em sẽ cố gắng cung cấp các ý về các bài toán của đề tài một cách dễ hiểu nhất. Thay mặt lớp, nhóm xin chân thành cảm ơn thầy đã chỉ dạy và hướng dẫn nhiệt tình trong học kì vừa qua.

1. **Giới thiệu**

Trong bài tập lớn này, nhóm sử dụng các phương pháp Black Box Testing để kiểm tra ứng dụng quản lí khách sạn được cung cấp.

Link project: <https://github.com/alandsilva26/hotel-management-php>



1. **Phân công nhiệm vụ**

Part A : Trịnh Duy Hưng – 1913652 – 100%

1. **Đặc tả ứng dụng**

Phần này sẽ bao gồm một số yêu cầu chức năng và yêu cầu phi chức năng của hệ thống mà nhóm tìm kiếm, phân tích và quyết định sẽ kiểm thử với black-box testing.

1. **Functional requirements**

Dưới đây là một số functional requirements mà nhóm đã tìm hiểu trên ứng dụng:

* Là một người sử dụng, tôi muốn đặt phòng..
* Là một người sử dụng, tôi muốn thanh toán online.
* Là một người sử dụng, tôi muốn đăng ký tài khoản.
* Là một người quản lí, tôi muốn thêm phòng mới.
* Là một người quản lí, tôi muốn chỉnh sửa thông tin của phòng.

1. **Non-funtional requirements**

Dưới đây là một số non-functional requirements mà nhóm đã tìm kiếm được:

* Tài khoản sẽ bị khóa nếu nhập sai thông tin đăng nhập quá 5 lần.
* Khi người dùng thực hiện thanh toán online, hệ thống không được phép lưu trữ thông tin thẻ credit/ debit của người dùng.
* Ảnh cập nhật phòng đúng định dạng hình ảnh.

1. **Kiểm tra yêu cầu chức năng**

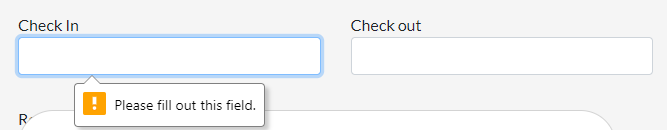
Các kĩ thuật Black-box testing để sinh testcase cho các yêu cầu chức năng của nhóm là:

* Boundary Value Testing
* Equivalence Class Partitioning
* Decision Table
* Use Cases Testing.

1. **Chức năng đặt phòng**

Chức năng đặt phòng cho phép người dùng đặt và sử dụng dịch vụ của khách sạn.

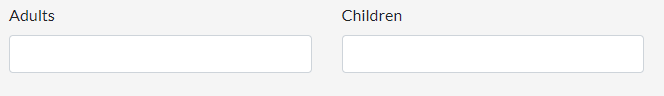
* 1. **Boundary Value Testing**
* Người dùng được yêu cầu phải nhập ngày đầy đủ nếu để trống.

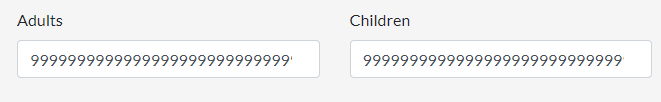
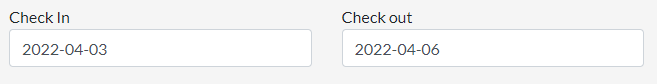


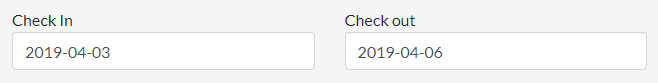
* Người dùng nhập số người là số âm vẫn được chấp nhận.



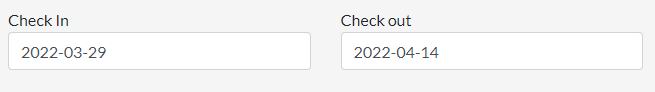
* Người dùng không nhập số người vẫn được chấp nhận.



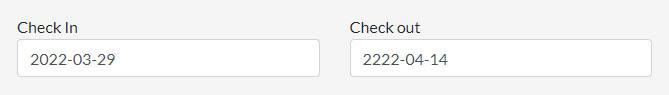
* Người dùng nhập số lượng người một cách vô lý cũng được chấp nhận.
  1. **Equivalence Class Partitioning**
* Ngày check-in và check-out trong tương lai được chấp nhận.
* Ngày check-in và check-out trong quá khứ được chấp nhận.



* Ngày check-in đến sau ngày check-out được chấp nhận.



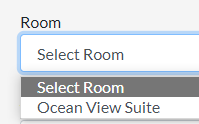
* Ngày check-out không hợp lý.



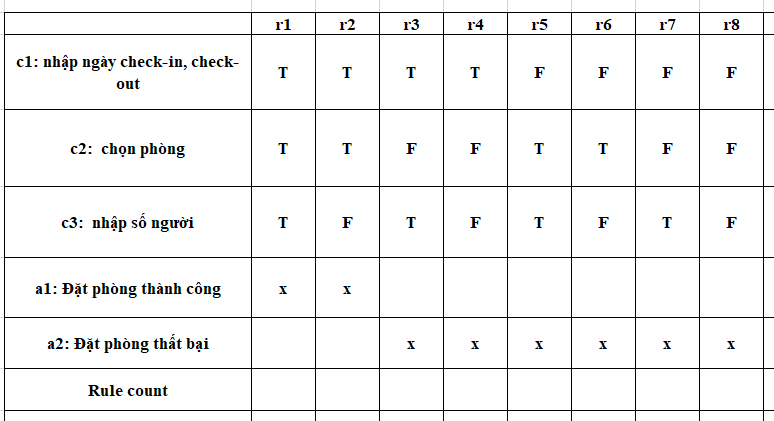
* Số lượng người hợp lý được chấp chận.



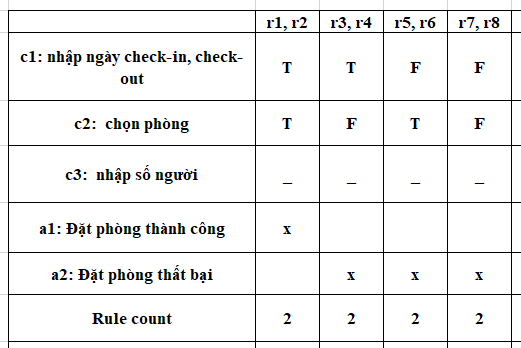
* Số lượng người không hợp lý được chấp chận
* Danh sách phòng được cập nhật sau mỗi lần đặt.

****

* 1. **Decision Table**



Từ bảng trên, ta suy ra:

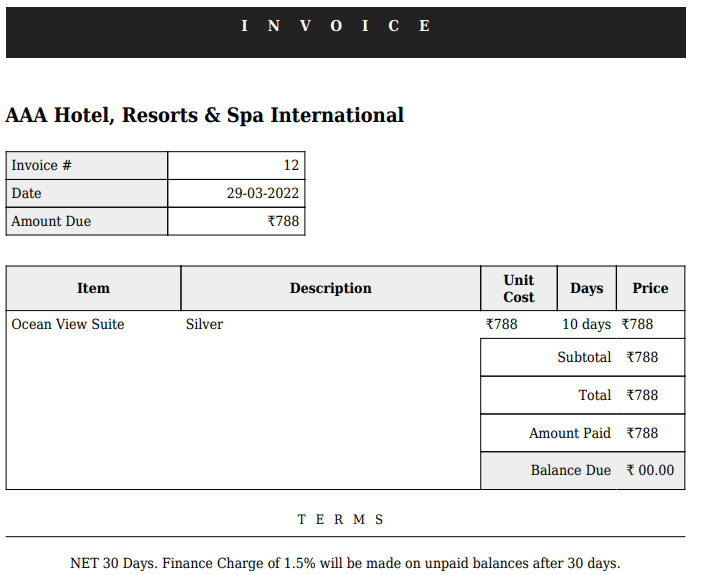
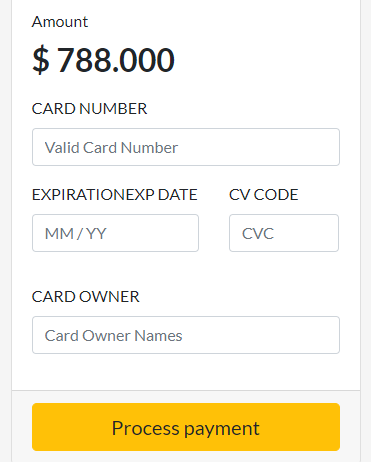


* 1. **Use Cases Testing**

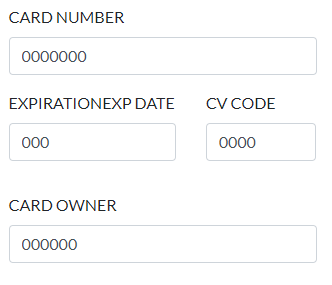
Các test case scenario của phương pháp use-case testing đã được mô tả trong file .xlxs.

1. **Chức năng thanh toán online** 
   1. **Boundary Value Testing**

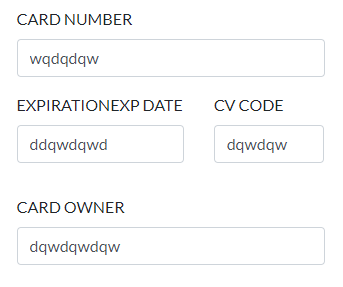
* Không nhập bất cứ thông tin gì nhưng hóa đơn vẫn được in.



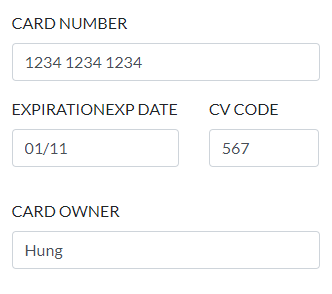
* Thông tin nhập không hợp lý nhưng hóa đơn vẫn được in.



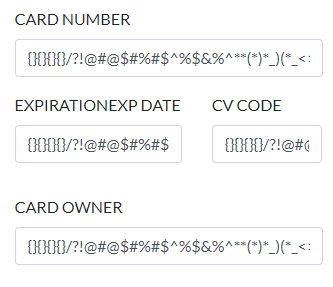
* Thông tin nhập vào là chữ nhưng vẫn được chấp nhận.

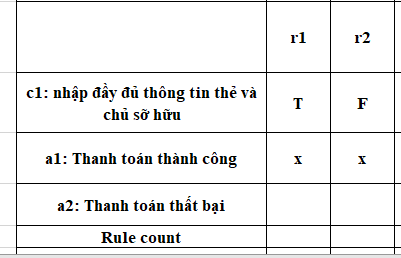


* 1. **Equivalence Class Partitioning**
* Thông tin chính xác được chấp nhận.

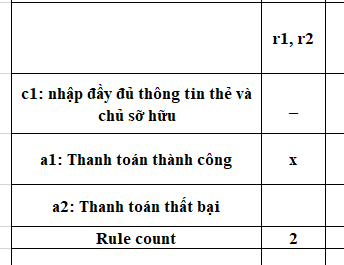


* Ký tự đặc biệt vẫn được chấp nhận.





* 1. **Decision Table**



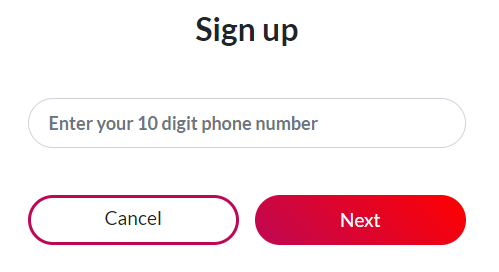
Từ bảng trên, ta suy ra:

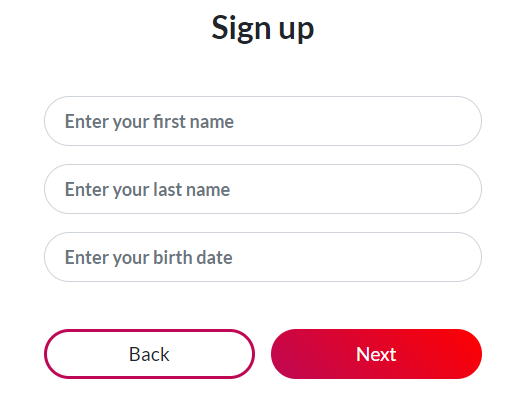
* 1. **Use Cases Testing**

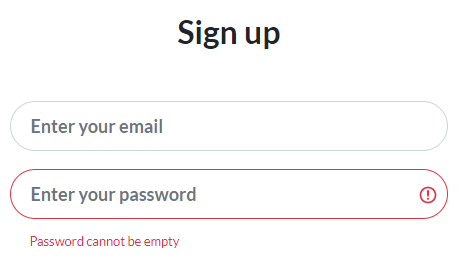
Các test case scenario của phương pháp use-case testing đã được mô tả trong file .xlxs.

1. **Chức năng đăng ký tài khoản**
   1. **Boundary Value Testing**

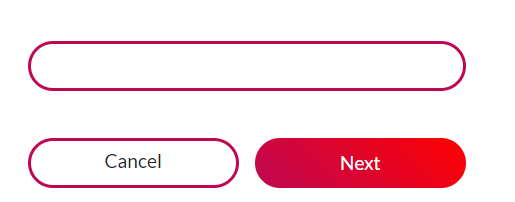
* Khách hàng không nhập số điện thoại, tên, ngày sinh, email vẫn chấp nhận, mật khẩu thì cần được nhập.

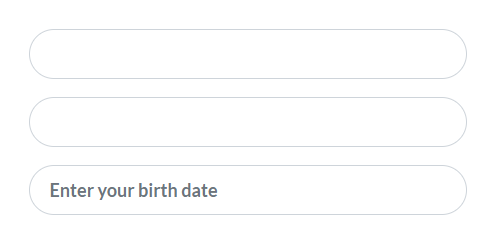


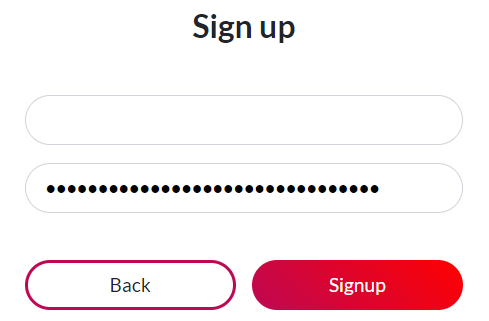




* Khách hàng nhập vào các khoảng trắng ở tất cả trừ birth date không thể nhập thì tài khoản vẫn được tạo và chấp nhận..



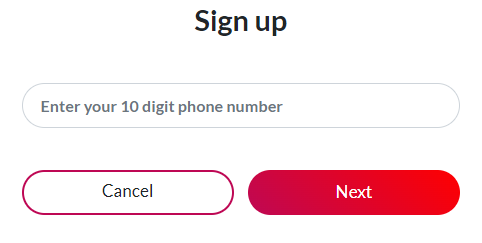




* 1. **Equivalence Class Partitioning**

Theo mô tả chuỗi nhập vào bao gồm 10 chữ số.

Dựa theo Strong Roubust Equivalence ta sẽ có 3 test case sau:



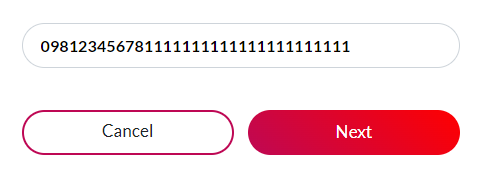
* Chuỗi nhập vào ít hơn 10 kí tự được chấp nhận.



* Chuỗi nhập vào đúng 10 kí tự được chấp nhận.



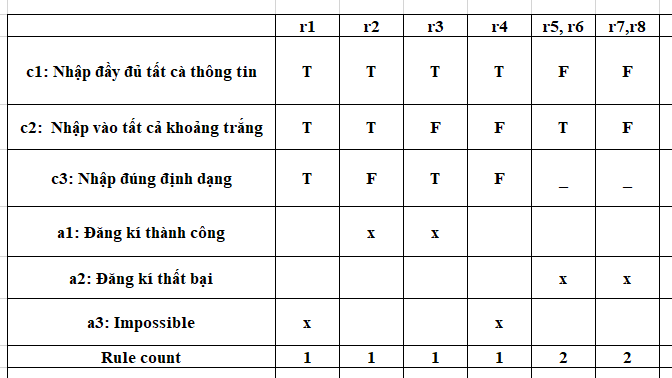
* Chuỗi nhập vào nhiều 10 kí tự được chấp nhận.



* 1. **Decision Table**



Từ bảng trên, ta suy ra:

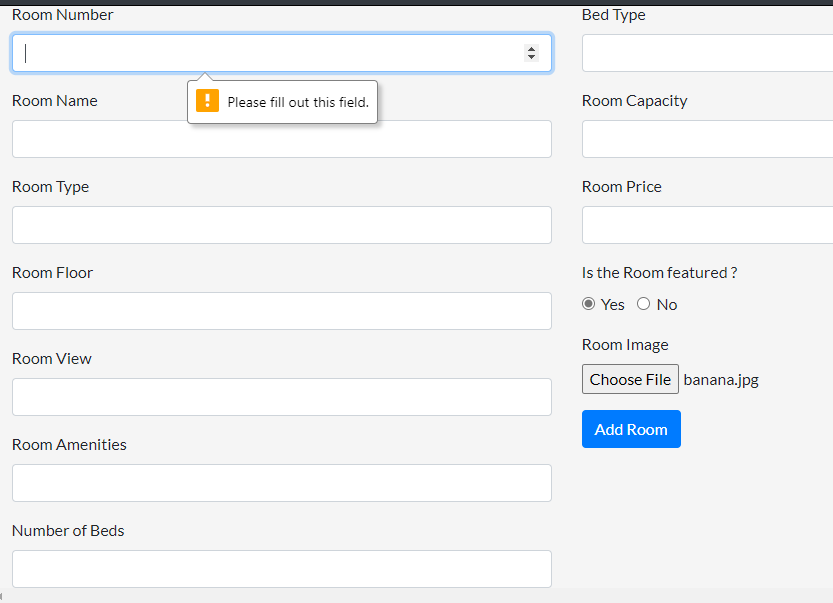


* 1. **Use Cases Testing**

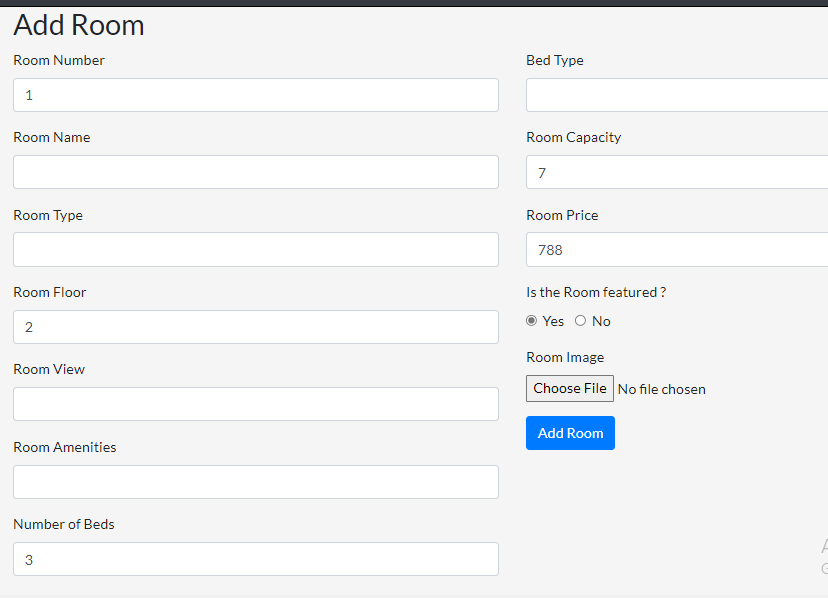
Các test case scenario của phương pháp use-case testing đã được mô tả trong file .xlxs.

1. **Chức năng thêm phòng**
   1. **Boundary Value Testing**

* Quản lí cần phải nhập đầy đủ thông tin, không được để trống và cần phải tải hình nền phòng lên để có thể tạo phòng.



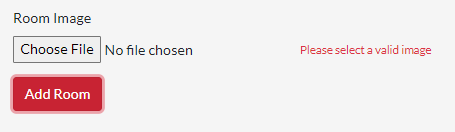
* Quản lí nhập vào những khoảng trắng và được chấp nhận.



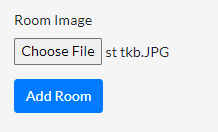
* 1. **Equivalence Class Partitioning**

Theo mô tả thì file hình ảnh bắt buộc phải tải lên.

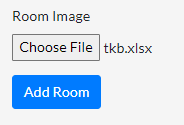
Dựa theo Strong Roubust Equivalence ta sẽ có 3 test case sau:



* Hình ảnh hợp lệ được chấp nhận.



* Thay file hình ảnh bằng những loại file khác cùng được chấp nhận.



* 1. **Decision Table**

Từ bảng trên, ta suy ra:

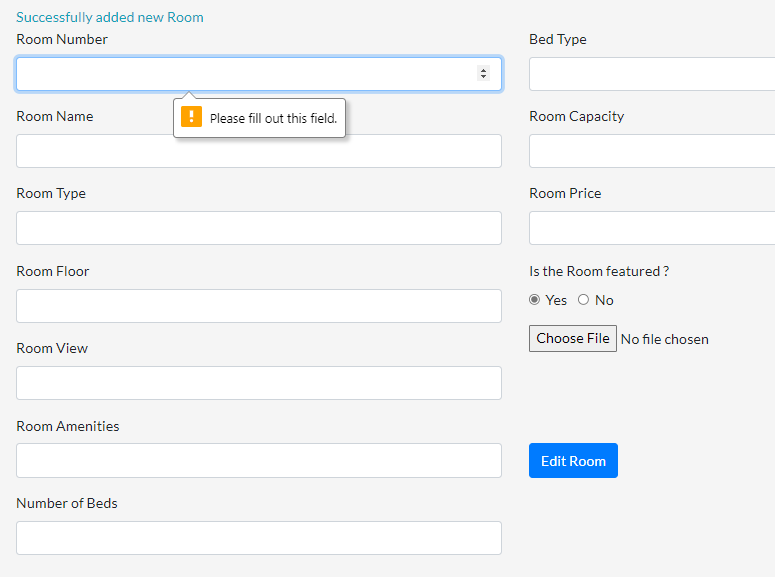


* 1. **Use Cases Testing**

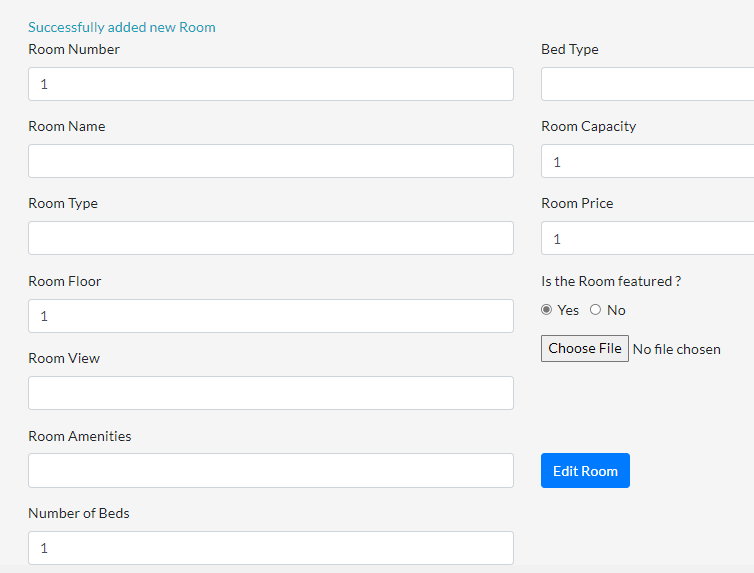
Các test case scenario của phương pháp use-case testing đã được mô tả trong file .xlxs.

1. **Chức năng chỉnh sửa thông tin của phòng**
   1. **Boundary Value Testing**

* Quản lí cần phải nhập đầy đủ thông tin, không được để trống và cần phải tải hình nền phòng lên để có thể chỉnh sửa phòng.



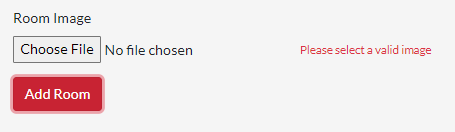
* Quản lí nhập vào những khoảng trắng và được chấp nhận.



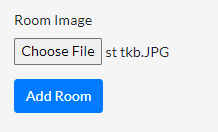
* 1. **Equivalence Class Partitionin**

Theo mô tả thì file hình ảnh bắt buộc phải tải lên.

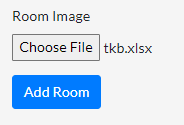
Dựa theo Strong Roubust Equivalence ta sẽ có 3 test case sau:



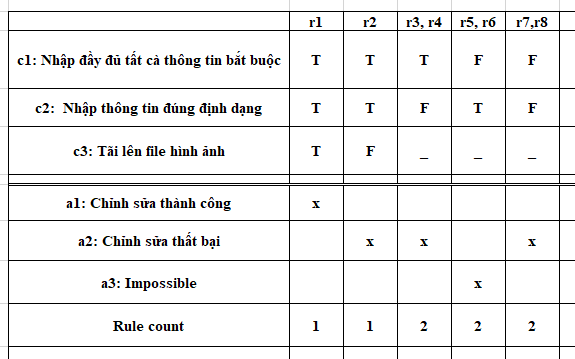
* Hình ảnh hợp lệ được chấp nhận.



* Thay file hình ảnh bằng những loại file khác cùng được chấp nhận.



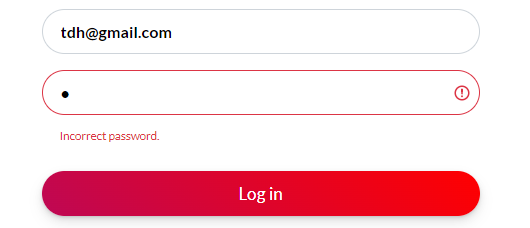
* 1. **Decision Table**

Từ bảng trên, ta suy ra:

* 1. **Use Cases Testing**

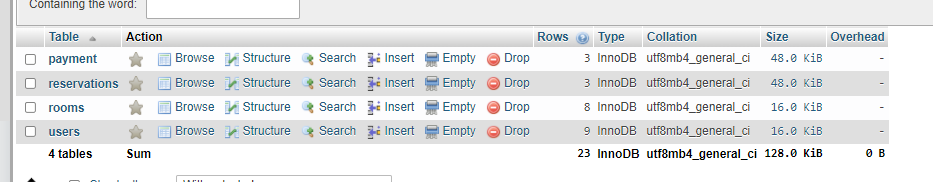
Các test case scenario của phương pháp use-case testing đã được mô tả trong file .xlxs.

1. **Kiểm tra yêu cầu phi chức năng**
2. **Tài khoản sẽ bị khóa nếu nhập sai thông tin đăng nhập quá 5 lần**



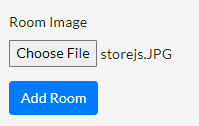
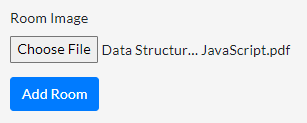
Ứng dụng không có cảnh báo hay giới hạn số lần đăng nhập sai.

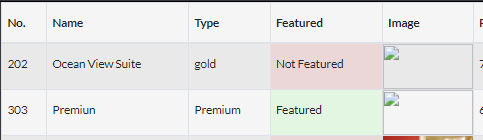
1. **Khi người dùng thực hiện thanh toán online, hệ thống không được phép lưu trữ thông tin thẻ credit/ debit của người dùng**



Khi người dùng thanh toán thành công hay thất bại thì không có bất cứ một thông tin nào liên quan đến tài khoản thanh toán được lưu lại.

1. **Ảnh cập nhật phòng đúng định dạng hình ảnh**





Khi cập nhật phòng thì hình ảnh được yêu cầu tải lên, những admin có thể lựa những file không phải định dạng hình ảnh để tải lên, từ đó gây ra lỗi hình ảnh.

1. **Tổng kết**