QA Methodology YouTube - TAMTECH Lineate Russiam

Test Design - planning and creating of Test Cases according to goals of testing and quality criteria. In order to provide the maximum number of bugs in an acceptable number of tests and in the allotted amount of time

Good Testing process - Is to create and run acceptable number of test that will represent STATE OF THE SYSTEM and find

most critical bugs To do it In allotted amount of time (short period probably)

**Test Design Techniques**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case**  Will test basic functionality  Will not test : every element separately  Will not test : different combination of entered data (text box)  Will not “broke” the system | | | |
| Who will be our USER?  (ROLE)   1. Registered 2. Not registered 3. Admin of group? | What are allowed actions?  Registered:  Send message, add friends, post on the wall, comment… | Especial options for different users?  \*Extra options for paying user  \*if you have many friends -you can see statistics  \*if man peoples are complaint about YOU-  Yore account will be monitored | TC 1-…  Every roll- every available action  TC 10-…  Every roll -every Not available action  TC 100-…  Every TC : a)with especial option  b)without especial option |

|  |  |  |  |
| --- | --- | --- | --- |
| **Cause and Effect** | | | |
| What is Input data?  What is output data?  What is the result? | FOR wall in Social Network:  Input :  User  Friends of User  Groups  Blacklist  Output :  Updated Wall of User  (User’s posts,  Friend’s posts  Group’s posts  NO posts of backlist) | 1  2 | Decision table:   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Friend? | F.in blacklist? | Group? | G in blacklist? | WALL | Pass/fail | | no | no | no | no | null | + | | yes | no | no | no | post | + | |  |  |  |  |  |  | |  |  |  |  |  |  |   Every one will be tested once at list |

|  |  |
| --- | --- |
| **State -Transition** | |
| \*State(status of user)  \*Transition(change of state)  #Event (action of user)->  #Action(reaction of system)  \*entry point(login)  \*exit point(save and exit)  Filtering/selection of test cases:  \*every state  \*all actions  \*every entry-exit way  \*every state-transition | List and order  Cancel ordering Make an order  Confirm and pay Cancel ordering  Successes payment Failed Payment Cancel ordering  Get ticket  Decision Table   |  |  |  |  |  | | --- | --- | --- | --- | --- | | State | Event | Action | New state | Pass/fail | | Selection of tickets | Filling the order form | Validation of filled data | Confirmation of the order |  | | Selection of tickets | Canceling of order | Deleting data about canceled order from DB | Canceling of order |  | | ….. | ….. | ….. | ….. |  | |  |  |  |  |  | | Buyer | Payment | Payment execution | Get ticket |  | | Ticket holder |  |  |  |  | |

|  |  |  |
| --- | --- | --- |
| **Equivalence partition**  (equvilalance classes / Boundary values) | | |
| \*if one makes bug-  Every one will  \*if one do not make bug-no one will not make  \*it tests takes values of same parameters (age and sex for example)  \*in executing of test the same blocks of script are  Launched  \*test are shows values of the same exit data | Divided into classes by:  Valid values  Invalid values  Divided into classes by :  Range of numbers values  Divided into classes by :  Fixed list of values(male, female ,boy , girl)  Variables with same value  (knowledge of the system is necessary)(same information entered in different parts of script like birthday data)  Time and date | **- ~**  **1 10 99 ~**  Allowed Number of tickets 1-10  **Equivalence class: -4 3 64 +12.5 + asddfd 115**  **Boundary values: -1 0 1 2 9 10 11 98 99 100**  **Positive 1 2 5 9 10**  **Negative -4 64 0 11 98 99 100 115 fhfh%%**  Not trivial classes and boundary: credit card  XXXXXX - ########## - Z  X - number of bank and type of card  #- unique personal number  Z - especial control number automatically created |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Pairwise testing**  (pair of two values) | | |
| Free online Program :  https://pairwise.teremokgames.com/ | Fields 3  Options for every field 10  Full count of tests:  10x10x10  1000 tests = 50 hours  Pairwise testing :  112 tests = 5.5 hours | Tickets ordering options to test   |  |  |  |  | | --- | --- | --- | --- | | DAY | TIME | COUNT |  | | Today | 13:00 | 1 | Maximum10 | | Tomorrow | 17:00 | 2 |  | |  | 20:00 | 5 |  | |  |  | 10 |  |   2x3x4=24 tests  Every test =3 minutes 24x3 =72 minutes  Pairwise testing -12 Test Cases   |  |  |  |  | | --- | --- | --- | --- | | # | Day | Time | Count | | 1 | Today | 13:00 | 1 | | 2 | Tomorrow | 17:00 | 1 | | 3 | Tomorrow | 13:00 | 2 | | 4 | Today | 17:00 | 2 | | 5 | Today | 20:00 | 5 | | 6 | Tomorrow | 13:00 | 5 | | 7 | Tomorrow | 20:00 | 10 | | 8 | Today | 13:00 | 10 | | 9 | Today | 20:00 | 1 | | 10 | Tomorrow | 20:00 | 2 | | 11 | Today | 17:00 | 5 | | 12 | Tomorrow | 17:00 | 10 | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Basic tests for start** | | | | | | | |
| Length: | minimum | AVG | maximum | Grater than max | Less than min | 0 |  |
| Text fields: | \*Numbers  \*Letters  \*Special signs  “[|]`~<!--@/\*$%^&#\*/()?>,.\*’ | Length of input | Spaces:  In Start  At the End  In Middle  Only space | Not ASCII  ⭕⭐ 😀 | Editing  Copy  \n \r end of string  Line beak |<- tab | Scripts:  SQL  `;SELECT \*  Html  <!--abc>  JS  <script>alert(“alert text”)</script>  Alert will be shown | languages |
| Number fields | The same like text  (see above) | + - 0 | whole/integral  Fractions(1.02,1/5)  Power of 2( 2 4 8)  Scientific (1E-10)  2+2 |  |  |  |  |
| Radio-buttons | Every option can be chosen | selected can be only one button | Can you leave none one selected ?  ( according to specification) |  |  |  |  |
| Check boxes | Every option can be chosen | All ch.b  Can be chosen | You can leave none one selected |  |  |  |  |
| Files | Size(see length above) | Type:  \*Valid  \*Invalid  \*content | Name:  Length  Language  Special symbols | Existence: of file (upload non-existent) | Edit /delete access |  |  |
| Date and time | Boundary of:  Minute  Hour  Day  Week  Month  Year  Centuries | Winter/Summer time | Non-existing  29 February  31 November | Time zones | Time /day from now? (does it matter it is past present feature)  Time depend on another time  (day of departure- day of return) |  |  |

GOOD TC:

TITLE: WHAT-THERE- WHEN

Steps

Expected result

Negative -positive

Agile :

People and communication > regulations and rules

Work product > documentation

Communication with Costumer > contract

Flexibility, reflection to Changes > plan