My apology to you because I am going against the theme of “Best Story” but I am obsessed with context driven testing ideas and in CDT we value good practices in different contexts and that’s why, I do not have a best story but yes there are some good stories in different contexts.

Before moving on, let me thank few people (who are my mentors knowingly or unknowingly) including James Bach, Michael Bolton, Jerry Weinberg, Pradeep Soundararajan, Parimala Hariprasad, Shrini Kulkarni, Lalit, Cem O Caner et. Al. Their work in terms of books, blogs, articles, tweets taught me software testing, quality, honesty, integrity, hard work, skills, and common sense.

These are different stories about my journey towards becoming a student of software testing and more importantly a tester belonging to CDT community.

My life in professional testing didn’t start by a well thought choice but my good testing stories are definitely an outcome of the choices, I made to learn software testing. The good part of these stories is about honesty and courage in exposing myself to the testing community I belong to.

Let’s begin

***Story about my first interaction with terms Customer, Desires, Money, Time, Quality, Templates (in software)***

In 2005, I started working as a bidder (a bidder basically place bids on software / IT projects on professional freelancing websites) in an IT services based company in India. I first came to know about few things including customer's desires (we labelled them as requirements), money they are paying to get that job of fulfilling those desires (budget), How magnificent they want it to be (quality expectations, yet always Implicit), How soon they need it (schedule). I also heard, read word ‘Quality’ but didn’t understand it well at that point of time.

I also heard about 'Template'. I will focus on this a bit more. I was given few templates prepared based on different kind of development, testing, SEO projects that our company has expertise on. My job was to look for projects on freelance website like elance, freelancer, guru etc. and place a bid by using the template. Just look at few keywords like {E-Commerce, shopping cart, payment gateway, NFL, sports etc.}, pick a suitable template and paste it as a response. Good to go. Almost no use of brain in understanding and connecting with customer desires, less thinking and less writing skills were involved (at least in my case, I am not sure about other bidders in my team).

Target was placing as many bids as possible and receive at least one reply either containing email of the customer or a contact number to be more helpful.

What I now relate it to read specifications (read project posted by customer) + write test case solely based on those (find key words from project posting and customize your template) + execute (post bid) + logging results (contact number, websites, email whatever once you get answer on bid)

One fine day, I was placing a bid on a project and before I could submit my bid, I saw it’s already placed by my company's CEO. For those eight lines 'Desires' where I had 20 lines in a template, my CEO replied just in 3 and the customer replied with his contact number and within a week we got that project. I immediately browsed for all bids placed by my CEO and it was just great. He knew his Business; he picked the very idea of what customer wants and instead of using template, he provided a crisp response. In my opinion He had ability to read emotions, desires of the potential customer from the written down text.

**My Problem**: I was hard working but in a foolish way, I didn't have a desire to ask question and hardly thought about the context of the desires posted by a potential customer.

**Learning**: Connect with the desires and hence to the potential customer, ask questions do not use template blindly place a competitive bid instead of lowest bid. You have to win customer confidence in as few attempts instead of just placing a bid passively

***Story about the way I started (misunderstood and abused Testing)…Excuse is “I didn’t know what testing is”***

My first experience with software testing was a practical assignment. A developer colleague came to me and requested....Hey, I developed an application which has a login control; can you test it? Imagine, knowing nothing relevant about software testing, I said yes off course, show me the application and I will test it for you .

**The way I tested**…

Opened the application, looked at login control, asked for a valid user name and password, entered it, hit login and I was on welcome Guest page. I said "Yeah, It's Working", it’s tested. That guy was embarrassed (I know you are too by reading this, but it’s true) and told me that I should have tried that login with invalid credentials to see if application stops me or not. I just shrugged and said okay. It took me some months after that to understand...

1. What that request actually meant (can you test it)?
2. How thoughtless my answer was (I said yes off course, show me)?
3. What was the model I was working with? Actually no model at all, no context, toe It was just a login accepting user name and password
4. What that guy expected me to do? Now I realize he expected me to ask questions, identify context, think, develop strategy, generate test ideas and then gradually start exploring & testing
5. What model he was thinking of? I believe a real world model of the login control w.r.t that single application and probably with others as well

**Learning** – A sincere response to the question ‘can you test it’ is an outcome of sincere thinking, learning, education, and practice in testing. Use your brain, think, question, model, setup context and respond accordingly. I didn’t do that then. But Now I do.

***Story about my desires, hands on and failures with coding***

I don’t know what was going on; some people say it was for good, I can’t say may be it was and may be it was not. After spending months, I decided to move to software development or programming. Why programming? It was not because I loved it; it was just because it was a respected job.

I requested to my boss to get me in to coding. I was told that I will be interviewed by programmers and if I clear the interview, I will be moved to 'development'. It was .NET era (I still remember purchasing ASP.NET 2.0 Unleashed) and reading it through days and nights. So the judgment day came, first interview, I didn't make it. I was frustrated but committed. I was given a second chance and this time I made it gracefully.

Did you notice the *words I used programming, software development and a shallow usage of "or" in software development or programming*?

Actually, to be honest, I passed a 'coding' interview. I didn’t care much about learning systems, technology, customers, business etc. Probably it was because coding itself is not easy.

Anyway, I was accepted as a 'New' coder and assigned an internal web based intranet application to work on. It was hard, very hard for me for initial days. Many times, I thought of being run away from that, but still I managed but it was late and at the same time I made a blunder. I didn't learn how to integrate client code to server code to HTML pages in order to build a running application, didn’t unit / integration / deployment tested and handed over to testing team. Nothing worked for that application, and project was handed over to someone else.

**Learning**: Learning how to code is important, but learning about system itself is more important e.g. who will use system for what purpose, underlying technologies, how these technologies interact, architecture, and design. Once you model system, you are able to program it and then apply your coding skills to make it work.

***Story of deciding to move into ‘Software Testing’ as a profession and journey towards becoming a student of software testing & part of context driven community***

Sometimes later on I felt that I didn't enjoy coding and I decided to move to software testing. As in case of choosing coding as a career did I love testing? No. I chose it because I thought testing is easy. 'Easy' was the keyword when I told my friend about this. My friend is a wonderful programmer.

He is a genius and told me right there that “…. you are just worried because you couldn’t do coding well, but you need to be more patient, practice more. Also ***software testing is not easy, it’s more challenging***….”

I had no formal education on software testing but yes I started reading ISTQB books (hold on…did I just said ISTQB, yes you heard it right but unfortunately its true). ISTQB was and still it is a most admired testing certification in India and many people suggested me to do that. You may be imagining how much crucial time I lost in digging stuff which was useless.

I saw people writing test cases in excel from a document in word, and then they execute those and write Pass / Fail, find bugs & log into in to a tool, done. With this thought process I requested my boss to move me to testing. To my surprise, nobody asked for an interview this time (unlike I was asked for coding), I said wow; it’s really easy, nobody is evaluating if I am eligible for starting career in testing. Anyways, I moved into testing and I started encountering words like requirements, SRS, frozen requirements, changing requirements, test cases, test plans, test strategy, automation, SUT, AUT, bug and so many words. So testing started (practically test cases writing and execution). No single thread of advice by anyone on heuristics, oracles, context, knowledge, skills.

Sooner, I found it difficult to write test cases and I still find it difficult, I liked writing tests more in a use case pattern rather than writing the procedural, step by step pattern based test cases. I started liking the ideas of…

1. Thinking about the software and its surroundings itself
2. What it does, what is its purpose, who are users, why are we building this?
3. Why are we talking about freezing requirements and so on?

I looked at test cases prepared by other people and I was pretty happy that I am on track. I write the same keywords ‘Verify that’, ‘User should be logged in’ etc. Since then I wrote a lot of test cases but I wrote in different fashions, one way of doing this work never clicked my mind, for example…

1. Step by Step precise steps derived from written and frozen requirements but with variable data points. This is sometimes suitable in certain processes like RUP
2. A generic one - A high level view of the entire test without going into details
3. A reusable one (sort of function calling another function) - linking on test case to another one and violating the idea of atomicity /uniqueness / independency of a test case
4. Automation scripts in TestCompleteTM - I enjoyed to see how a machine mimics the user’s actions and amazingly after this many years I still see that we are struggling with 'Object Recognition' :)
5. Now, I am an interested in generating beautiful test ideas through discussions, questioning, exploration, and present them through mind maps, whiteboard, wiki and other suitable lean methods.

While I was doing all this, I joined another organization and ***this was a turning point in my career***. I found the new environment bit different from past. To emphasize on few things including fast paced development, a dedicated testing team, product documentation but focus on lean test cases, configuration support team, release management teams and delivery etc.

Here is where I actually started learning the 'exploratory testing’. I tested the application like I wanted to crash it. I was happy that I was gaining confidence in plunging into using and checking the software against specifications. I found issues while exploring applications in order to understand it better. Let me tell you this…largely it was an unguided exploration but it worked because at least my brain and my test choices guide this exploration but it was not effective I must say because of lack of good oracles and heuristics knowledge.

***Expanding exploratory skills, learning regression testing, reverse engineering, product coverages and automation***

We were assigned the testing of a stable and existing financial product with expectations that we would explore it, learn it and gradually start test it. Test manager approached to us and told to soon new development will start on this product and we also need to regression test this. All we had were few test Cases written by SMEs, a user manual and few use cases. My exploring skills actually flourished on this testing assignment. Some activities which helped me in testing this product effectively…

1. I browsed old bug database to understand what problems SMEs found earlier and how they represent impact,
2. How they decide priority of fixing a bug?
3. Explored and tested other applications within the same project umbrella with focus on bug hunting
4. Product configuration and integrated systems configurations
5. Exploring and creating knowledge repository for the team, giving product sessions to business analysts from competency centre
6. I learnt about banking, deposits, lending solutions and vast variety of customizations for different customers
7. I first time interacted with CMMi groups who were responsible for quality assurance related processes
8. I got married (It is also a good story but out of context :)).

This assignment was full of opportunities to learn, explore, and interact with different groups and so on. Few things are worth to mention

1. I learnt the product by ***teaching the product***
2. I carefully ***listened*** the talks on testing within company
3. ***Observed*** how different people tests a single piece of software
4. How they are ***motivated to explore***
5. How they relate a function to a business need
6. I learnt how to do ***constructive arguments*** as my colleague in CMMi implementation group was eager to listen to my views as I was opposing useless and irrelevant processes in my project context
7. I did a reasonable research on combinatorics problem and exercised tools like PICT (from Microsoft), ACTS from NIST (Thanks to Dr. Richard Kuhn)
8. I learnt tools like QTP with extensive self-study and hands on practice sessions days and nights

***Certifications, Books, Blogs…how all it started?***

This was the high time on ***learning, motivation***; I was attracted to ISTQB, CSTE, ISEB and all such other certifications. Clearing certifications were like wow factor for me. I just cleared those without figuring out if they are really helping me or not? In my experience it was just ***passion*** to learn.

I re-started reading books (after college). I purchased almost every book for ISTQB but hardly read any of them for more than one or two chapters. I found those too mechanical; this was the time when I first encountered…

1. A blog by B.J Rollison and especially an article and related debate on equivalence class partitioning
2. Exploratory Testing book by James Whittaker.
3. Software Testing club blog

I found all of these interesting to read. Especially, I read the book carefully and tried to apply some concepts of it in my day to day testing. I found it bit difficult, you know why? I think it had to do something with tacit knowledge. In my opinion, when James W was writing this book, his tacit knowledge of testing and extensive work on touring analogy must have been prominent in while putting down that knowledge to words and hence though I found it interesting but at the same time found it difficult to implement the knowledge in my project. But this Book purchase led me to purchase few more books on 'How to Break..." series from James W. These were interesting read and by this time, I realized that my testing knowledge was shallow; it was linear and was just moving around requirements, bugs and test cases.

***Fortunately accidentally***, one day, I found the gem, [***Lessons Learned in Software Testing***](http://www.amazon.in/Lessons-Learned-Software-Testing-Context-Driven/dp/0471081124) (in pdf format from internet)...Here was my first thought… ***Hmm, seems another James, may be this book won't help me much like James W’s book didn’t***. So I almost neglected it by reading the cover page saying 'A Context Driven Approach'. What the hell is Context? I randomly accessed my memory and found that "I never heard about Context" so I was biased and decided not to read it!! But after few days I started reading it but casually and didn’t take it seriously for few next days (the same way during last year when I read Jonathan Kohl’s classic read ‘Tap into Mobile Application Testing’). I purchased other many books on software testing techniques, strategy, and automation and read those and then the time for ‘enlightening’ came….was it late?, yes, it is late, but still It came to me and I made a good choice.

I was considered a good, hard working, passionate tester and explorer. But here what I evaluate now about myself.........?

1. I was testing well but I was not aware of context driven testing. I probably was doing testing which was context specific or context aware, but it was not context driven.
2. I lost interest in writing test cases unless I was really required to do by the law of manager or client. I was more on exploring and reverse engineering side
3. I did test planning on IEEE829 template which contained a section called strategywhich I think it was taken granted as it was usually a static text and was never looked upon by anyone. It was also too much repeated and even no tester from my team ever looked at it once I written it down and deliver to my manager. One fine day I thought about it and went to a more experienced tester to ask can you tell me the purpose of this document? How to make a difference between test plan and test strategy. Since I did my homework already (looked at test plans from other projects, googling, reading books, blogs etc. Alas! During my homework I didn’t open ***Lessons Learned in Software Testing***), anyways, I clearly told him that 'Please don't play with words' …I need a clear answer, I think he started his answer with word 'Document' and I fell down :).

This was the time when I contacted James Bach over an email with my confusions on the definitions and usage of test planning and strategy. I provided him the details of my homework and I think he replied me with the answer that he found me enthusiastic and we got connected on skype. He asked me some questions and then asked do you have my book lessons learned in software testing? Did you read its last chapter on strategy and planning?

I immediately opened that and literally cursed myself for loosing precious time in searching here and there instead of directly look into this book***.*** This book, from this point of time in my career, completely changed the way I thought about Software testing***. My fellow Story readers buy it now.***

***And finally this is the Story I am most enthusiastic to share***

Few lessons I am obsessed with (though this book is a coal mine), are worth to mention and those helped me in re-shaping my thoughts on testing

1. *Studying epistemology helps you test better*
2. *Use the logic of Conjecture and refutation to evaluate a product*
3. *You can’t master testing unless you reinvent it*
4. *Bug Advocacy*
5. *Planning the test strategy*

I understood the power of testing skills, thinking, questioning, and reasoning. I invested in testing literature recommended by James. This was and still is a beautiful time in my life, in the sense that I feel myself now a student of software testing who is making progress in right direction. This was the time I came to know about classics from [Jerry Weinberg](http://www.geraldmweinberg.com/Site/Home.html) like [*Exploring Requirements*](http://www.amazon.in/Exploring-Requirements-Quality-Before-Design/dp/0932633730) and [*Perfect Software*](http://www.amazon.in/Perfect-Software-Other-Illusions-Testing/dp/0932633692). At the same time I am glad to mention about the work of [Michael Bolton](http://www.developsense.com/blog/) and his assistance on resolving some of my queries over the time (sometimes on email and sometimes on Twitter).

While LLIST and reading carefully [James Bach’s blog](http://www.satisfice.com/blog/) made me more sincere about ‘software testing’ , studying books from Jerry made me more curious about knowing ‘systems’, ‘requirements’, ‘quality’ and obviously on ‘testing’. To be very honest, in first place, I was not comfortable reading Jerry’s books; they always give me a feeling of ‘learn basics first’. I often ask myself, why I missed these books in my college time and even in starting of my career’.

I continued looking for more knowledge and read blogs from [Shrini Kulkarni](http://shrinik.blogspot.in/) and again I found an amazing tester to interact with and I do often now a days. Recently I discussed with him about my certain queries on ‘context determination’

I also got opportunity to meet (personally and in a conference) the ‘Panda’, [Pradeep Soundararajan](http://testertested.blogspot.in/). He helped (healed) me in on different aspects of professional and personal life

In all, my professional testing skills started building just few years ago, before that it was not of that level and I am still learning but as I said I think am in a right direction. The good thing is that I started understanding ***Power of thinking in testing***

In this story, one book is worth mentioning which is “[Celebrating Silence](http://www.amazon.in/Celebrating-Silence-Sri-Ravi-Shankar/dp/8190796402)” by Sri Sri Ravisankar (a well-known spiritual leader). This book helped me to discover a hidden and often overlooked aspect of human beings ‘Enjoying Silence’. To me it’s a form of meditation and calmly observing what’s happening around and responding to the events which are important. It helped in testing as well, observing and taking out what is important at that point of time.

For me, when I read work, listen to from testers in context driven software testing community or other good testers around the globe who may or may not be in this school of thoughts. I keep on improving on my thinking and also try to add more skills.

How can I end this story without mentioning Automation experience? I did automation of regression checks and probably I was Context oblivious and I always had in mind that its only regression checks those should be automated. But [Alan Page](http://angryweasel.com/blog/) who wrote a beautiful book on automation “[The A Word](https://leanpub.com/TheAWord)”, changed my perspective on automation.

*C****ourses, Classes and Conferences (3C’s)***

Once I became aware of ‘***What really software testing is***’. I decided to move fast and consistently in right direction without taking a break. I invested into purchasing recommended books, attending courses, online classes and attending conferences. I took [RSTA](http://www.satisfice.com/rapidtestintensives.shtml) from James. If you are reading my story and you are a software tester who didn’t take this class yet. Don’t waste time…DO IT RIGHT NOW and learn from the Master.

[BBST Foundations](https://www.associationforsoftwaretesting.org/training-2/courses/foundations/) was another course that opened my eyes. Believe me or not. In my experience, this is one of the finest ways of learning our craft. You get exposure to express your ideas, improving upon your thinking and writing, Review skills and honestly evaluating yourself. Take it! We can learn a lot from this course.

Specific to conferences, I had very less chances to visit. My first conference was BWST in Bangalore (I met [Parimala](http://curioustester.blogspot.in/) there, an awesome tester and organizer) and then ThinkTest in New Delhi organized by charming lady [Smita Mishra](http://www.qazone.in/).

Probably I will speak in some conference someday☺. I actively take part in week end testing sessions. I remember when I usually said to myself ‘I am bored of testing’ and now I find myself how less time I have to get bored. I don’t have time, really.

This story, especially the last one, has a number of testing heroes, heroines whose work in any shape helped me to understand systems, quality, testing, collaboration etc. better. This story is not ending here as it is the better story than previous ones, in the sense that I know foundation of my testing was not that good but now I decided to re-look at foundation to build a new one.

***Tweet, tweet…***

I joined twitter that opened whole lot world of learning opportunities, connecting with different testers around the globe and believe me 24 hours in a day are less for the vast knowledge scatted in this space through different blog posts, papers published, testing quotes, archive information, new ideas and so on.

This is my story about a journey on becoming a student of software testing and knowing nothing to knowing something about Software testing.