HI!

I am Miki Szeles from Hungary, Budapest. This is my debut here on the Test Project blog.

I have my friend DJMic with me. He is responsible for spicing up the mood of our readers. DJMic! What do you have for us today?

DJMic: HI, dear readers! Today we have 3 rooms from which you can choose:

1. The focus room: Just [click here if you want to hear classical music,](https://msz.team/3vjqMrY) which helps you concentrate!
2. The world-class rising star: [Click here if you would like to listen to a world-class Hungarian band called The Carbonfools!](https://msz.team/3KERSjD) At the drums, [Benedek Hámori (IT) Recruiter](https://msz.team/3M2VpZi).
3. Nostalgia stage: If you are also from the 80s like Miki, and [if you would like to relive your grammar school years, just click here!](https://msz.team/3xKGa3x)

Let the party start, Miki!

Miki: Thanks, DJMic!

## Who am I?

I am a father, husband, and soon to be a happy dog owner.

I am an agileish technicalish writer working together with more than 30 creativeish friends to provide the best experience to you.

I have 15 years of professional experience in Software Development, of which 8 also includes leading the dev team and 1 year included leading the test team.

I am an Agileish Superhero with the superpowers of super learning, super thinking and super memory.

If you want to learn more about me, just read my bio!

## Why reloaded?

* Minnie: Miki! We have already written an article with the name To Test, or Not To Test, That is Not The Question. So what is this?
* Miki: First of all, it was only me who had written that article. Only me.
* Minnie: You are right. And?
* Miki: And it was much more boring than the ones we wrote together.
* Minnie: 😊. So why now?
* Miki: McRuiter just found a new partner for us.
* Minnie: Really? Who?
* Miki: He is called Nester!
* Minnie: Hm. I like the name. It is like a jester. Is he the funny guy?
* Miki: Nope. We already have Nikolai and Mish for that.
* Nikolai: F\*ck off.
* Mish: Thatish veryish kindish fromish youish. Thankishish.
* Minnie: So then, from where the name Nester is coming from?
* Miki: From the word tester. Nester is the son of NoSEO and the twin brother of Neveloper.
* Minnie: Fascinating, but I know that family. They are the worst experts in the world.
* Miki: You are right, Minnie. You are right.
* Minnie: Then why Miki? Why?
* Miki: The only thing you have to do is exactly the opposite of what they suggest.
* Minnie: Wow. That is a great idea.
* Miki: I know. I know. It was mine.
* Minnie, Nikolai, McRuiter: It was mine.
* Miki: Ok, guys. As you like it. It was ours.

## The right question is: How much of our humanless automated tests should be converted to humanish manual tests?

Minnie: Hmm. Interesting, I would suggest following the structure of the previous article, so the first question is:

## Why should you test manually?

### The cost of test automation

* McMuck: Everybody knows cost efficiency is one of the USPs of test automation.
* Miki: Well, I am not good with numbers, so our expert Manality will do the calculation as usual.
* Manality: We live in the remote work era thanks to COVID, so to find a good expert, you have to compete with all the companies worldwide. As all companies are looking for test automation engineers, [the average yearly salary of a test automation engineer is $95,149/year.](https://webcache.googleusercontent.com/search?q=cache:AFTJc4rvKNcJ:https://www.ziprecruiter.com/Salaries/International-Automation-Engineer-Salary+&cd=13&hl=hu&ct=clnk&gl=hu) India has the second-largest workforce globally. There are 501 million workers, which means the workforce available in India is 50 as much as the total inhabitants of Hungary.  
  [In India, the average yearly salary is ₹382,500](https://khatabook.com/blog/know-about-average-salary-and-wage-in-india)/year which means $4994/year. This means you can employ 95149 / 4994 = 19 manual testers for the price of a test automation engineer.
* Miki: Thanks, Manality! So let me repeat it, **you can employ 19 manual testers for the salary of one test automation engineer**.
* Nester: It is clear you have to go with manual testers. As you can see, **they can do 19 times more work than a single test automation engineer for the exact cost.**
* Miki: Yeah, but you know, you only have to write automated tests once, and you can run it forever. You do not have to give food to it.
* Nester: Well, that is another USP of test automation, but is it true? Have you had to touch the already written test automation code?
* Miki: Well, I have to admit, yes.
* Nester: Why?
* Miki: Most of the E2E tests are flaky due to timing issues. I heard about a company where cookie banners appear randomly multiple times, one after the other, just to mention an example.
* Nester: Anything else?
* Miki: Sure. Often, texts on the UI change, so if I want to make sure the correct texts appear, I have to write tests for that, but they break on a single translation change. Not even mentioning there can be dozens of supported languages.
* Nester: I see. Anything else?
* Miki: Sure. The most significant source of change is the change in functionality. In today's agile world, instead of designing and implementing a full-fledged feature, we only implement featureish, the bare minimum that gives the most value to the customer.
* Nester: Ahh, agile. That is worth another story. So what does changing features mean in terms of testing?
* Miki: You have to adapt each test after changing the functionality.
* Nester: I knew it. Miki, have you noticed our data expert Manality calculated the average salary of an Indian person instead of the maximum wage, similarly to what she did with the test automation engineer calculation?
* Miki: Yeah, actually, I wanted to ask about it, as it has been on my mind since I saw that calculation.
* Nester: The reason is straightforward.
* Miki: What is it?
* Nester: Anyone who can use a browser can test 99% of the applications currently available on the market.
* Miki: Makes sense, but in that case, we could calculate with a salary close to the bare minimum.
* Nester: We could, but we are a humanish team, so we won't aim for the minimum salary.
* Miki: Makes sense.
* Nester: That is enough about the cost of testing. Let's examine testing from a theoretical point of view.

### Beware of the pesticide paradox.

* Nester: Miki, I know you have an ISTQB CTFL test, so I guess you do the testing fundamentals, including the 7 testing principles.
* Miki: Sure. I even wrote an article about it.
* Minnie: Really? I have never seen that?
* Miki: Yeah, because that was never finished, it is still amongst the 20+ drafts I have unfinished due to changing priorities.
* Minnie: Oh. I see. Thanks.
* Nester: So, Miki! Please explain what the pesticide paradox is.
* Miki: The pesticide paradox brings an analogy from the real world: The more pesticide you use, the less effective it will be over time, as the insects adapt to the pesticide.
* Nester: I see. Now, let's hear the official mumbo jumbo.
* Funality: This term comes from [Leandro Melendez's The Hitchhiking Guide To Load Testing Projects: A Fun, Step-by-Step Walk-Through Guide](https://msz.team/3xoRS3w) book.
* Miki, Minnie, Nester: Who are you, Funality?
* Manality: He is my younger brother, he is also obsessed with facts, similarly to me, but he is the "funny" guy in the family.
* McMuck: Cool. This McRuiter knows what he is doing. I smell money here. We will write a fat 50$ bill for each fun fact coming from Funality.
* Miki: The official Mumbo Jumbo is the following: The more you repeatedly run the same test suite, the less likely it will find any bugs in the code.
* Nester: You see! All the benefits you can get from test automation significantly degrade over time.
* Miki: That is true.
* MatchyT: I would call it the Inflation of Test Automation!

### The Inflation of Test Automation

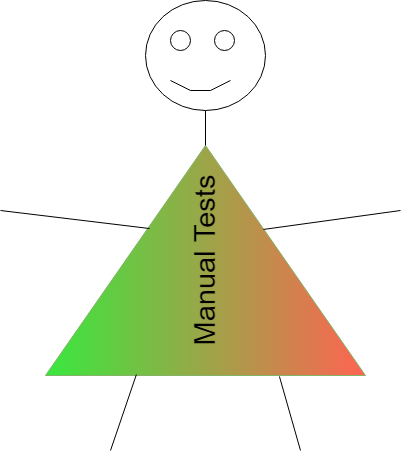
* Miki: Who are you, MatchyT? I have never met you before.
* MatchyT: I was always here in your head, Miki, but finally I got a name, and I can speak on my own, so it will be me, the one who really deserves it, who will get the appreciation for coming up with catchy titles and terms.
* Miki: I see. So do you think "The Inflation of Test Automation" is a catchy phrase?
* MatchyT: Yep.
* Miki: We will see. Ok, so how can manual testing address the problem of the inflation of test automation?
* Nester: Simply, by letting humans do the work.
* Miki: But humans are making mistakes which is not valid for automated tests.
* Nester: First of all, humans are not making mistakes; they just examine things from different perspectives depending on their current state of mind. Secondly, that is what makes sure the value of your tests won't degrade over time.
* Miki: I do not get it. We have clear step-by-step test cases. Even if testers make some mistakes, they repeatedly execute the same steps.
* Nester: Miki! Why do you think we need test specifications?
* Miki: To make tests repeatable and to make sure we test precisely the same thing again and again.
* Nester: Do you see why this leads to the pesticide paradox?
* Miki: Nopeish.
* Nester: If you test precisely the same thing repeatedly, you won't find new bugs with it.
* Miki: So what should we do then? How can we live without test specifications?
* Nester: There should be one single source of truth.
* Miki: Are you talking about the specification?
* Nester: Almostish. I am talking about the user story. The user story should be the single source of truth,
* Miki: Why?
* Nester: First of all, that defines the customer's need. That is what the customer really wants to have.
* Miki: Ok. What else?
* Nester: That is based on which developers should develop the feature, and that is based on agileish testers should test the implementation.
* Miki: What if the user story is not clear enough?
* Nester: In that case, you have to make it clear enough. You can resolve the neverending debate between developers and testers by doing that.
* Miki: Which debate?
* Nester: The debate of "It is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a feature, not a bug! F\*ck, you idiot! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! You son of a b\*tch! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a feature, not a bug! Are you still reading? No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! This is the fault of that idi\*t manager! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug! No, it is a bug, not a feature! No, it is a feature, not a bug!".
* Miki: Nester!!! Stop! I got the point! What is the solution, then?
* Nester: The solution is simple: If something is not in the user story, **it is a bug.**
* Miki: But what if it is valuable?
* Nester: Then it should go into the user story.
* Miki: I have to admit, it makes sense. I have wasted many hours of my life and my employer's time too by takinparticipatinghis Feature vs Bug debate.
* Nikolai: You are weak, Miki. You are very weak.
* Miki: Thanks, Nikolai. I can always count on you.
* Nikolai: It was my pleasure, Miki. It was my pleasure.
* Miki: Nester! What about continuous integration? Without that, you cannot deliver in our fast-paced world.
* Nester: You are right. Let's talk about the Agileish Humanish CI/CD pipeline (credits go to MatchyT).
* Miki: About what????

### The agileish humanish CI/CD pipeline

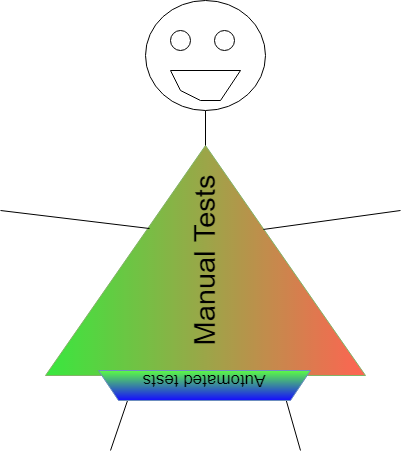
* Nester: About the agileish humanish CI/CD pipeline
* Miki: This sounds crazy, but you made me curious.
* Nester: So, what is the main point of the CI/CD pipeline?
* Miki: To have immediate feedback.
* Nester: Exactly. It is needed to ensure developers do not have to wait a long time to see whether the changes have broken the build.
* Miki: Yep, I believe a short feedback loop is the most critical factor in improving anything. That is why I made my whole life agile.
* Nester: Your whole life? How?
* Miki: Yep. Just read my [How You Can Become An Agileish Super Learner AKA How You Can Become A Super Thinker AKA The Proof Of Concept (POC)](https://msz.team/3vEc4Mr) article.
* Nester: Okayish.
* Miki: How can you get immediate feedback by doing manual tests?
* Nester: It is simple. As soon as something is ready, manul testers can test it immediately.
* Miki: But what if the tester is busy?
* Nester: You still have 18 manual testers who can do the job.
* Miki: Ok, but what about timezones? What if the team is distributed and working from different time zones?
* Nester: Let's split the day into six 4 hours blocks.  From the 19 testers, you can form 6 groups, all of them having 3 testers, and you will still have one extra available tester, which might come in handy if somebody is sick.
* Miki: Ok, so testers should work for 4 hours?
* Nester: Well, that is an option too, but you can decide whether you want to have testers in 4, 6, or 8 hours per day, depending on the needs of your company and the needs of the testers.
* Miki: Let's stay with the 8 hours per day, which means at every single point of time, there will be 6 available testers. So this means they can test 6 features parallelly.
* Nester: That is true. But let's be realistic. If a feature needs 4 hours to develop and 4 hours to test, then it means 1 tester can keep up with the pace of one developer. Let's suppose the company recognised the importance of testing, so they have employed one single test automation engineer.
* Manality: In Miki's team at DMG MORI HEITEC Digital Kft. they have 1 + 2 \* 0,5 = 2 test automation engineers and 1 + 2 \* 0,5 manual testers. This means we have the budget to employ 2 \* 19 + 2 = 40 manual testers. The dev team is around 10 people, so there could be 4 manual testers for one developed feature.
* Nester: Yep. That is 4 different viewpoints for each test, which is way much more valuable than you can get by automated tests.
* Miki: Do you want me to be fired?
* Nester: Nope, you also have your part in the equation, but let's talk about that later.
* Miki: Ok, so what about regression testing? Any single change in the code can break anything. You know, many companies have **neural code** bases.
* Nester: Neural code? I have never heard about that.
* MatchyT: I came up with the name. It is straightforwards. The code pieces are like neurons in your brain. **Almost every part of the code is connected with nearly all the others.**
* Miki: Oh! So that is neural code.
* Nikolai: I have never seen such code in my professional career.
* Miki: Nikolai! You do not have something called a professional career. You cannot even materialise yourself.
* Nikolai: Believe me, Miki. Without me, you would be serving french fries in a McDonald's.
* Miki: Well, you might be right. Anyway, I have to admit, I have written neural code multiple times during my life.

* Miki: So, let's get back to regression testing.
* Nester: Regression what?
* Miki: You know. Regression testing is when testers test whether the non-changed features are still working.
* Nester: Oh, you are talking about that. Well, **manual testers should not do regression testing apart from the primary usage of the application**.
* Miki: Wait, what????
* Nester: **If developers write clean code, following the single responsibility principle, there should be no side effects that the developer does not test.**
* Miki: But how would a developer know which non-touched part should he check?
* Nester: Let me throw the ball back. How should a tester who does not even has access to the codebase know what might go wrong as a side effect of the change done by the developer?
* Miki: Fair enough. So what can we do in this situation?
* Nester: The **main problem** is that most **companies are doing brute force regression testing.**
* Miki: Brute force regression testing? What do you mean by that?
* Nester: They are **not trusting their developer's expertise**, so **they test everything** instead of trying those one-two parts to which a developer can narrow down the potential side effects.
* Miki: What do you think can be done here?
* Nester: We should not answer all the questions. Let's give some homework to the readers
* Miki, Nester: Dear Readers! How can we narrow down the scope of regression testing knowing the exact lines which have been modified?
* Miki: Let's move on to the test pyramid.
* Nester: Sure.

## The Humanish Test Pyramid

* Miki: Humanish test pyramid? Never heard of that.
* Nester: Of course not. I have created it.
* Miki: Interesting. How it looks like?
* Nester: It looks exactly like this:
* Miki: So you say that no automated tests are needed?
* Nester: I have never said that.
* Miki: Then where is it?
* Nester: You know, the first versions always s\*ck. The whole point of existence is to have something on which you can spit.
* Miki: Yeah, that is true. But they are a necessary bad. As soon as you have the first version, you can start discussions about it.
* Nester: Exactly.
* Miki: We went way too far with this article; how can we make sure I can keep my job as a test automation engineer?
* Nester: Don't worry, Miki. I already thought about that.
* Miki: So where is test automation still needed?
* Nester: **Test automation is needed to create the preconditions for manual tests.**
* Miki: Phew. I am saved. That makes a lot of sense.
* Nester: I know. I came up with it.
* Miki: What about tedious, boring tests? They have to be automated, too, right?
* Nester: Nope.
* Miki: Why????
* Nester: If a feature is tedious/boring to test, it is tedious/boring for the user too. It is a design smell.
* Miki: Can you explain it?
* Nester: Sure. If you have something too complicated or complex to do, you probably do not have a UX designer.
* Neveloper: Of course, we do not need a UX designer. Every developer can create a logical, straightforward UI.
* Nester: HI, Bro! Is this logical to you or to the user?
* Neveloper: Logic is human independent.
* Nester: Are you sure about that?
* Neveloper: Of course, pure logic is the only thing that matters.
* Nester: What about emotions?
* Neveloper: Emotions have nothing to do with logic.
* Nester: Well, I would argue that. Don't you think users have emotions?
* Neveloper: Of course they have. They are humans.
* Nester: And do you think they will have the same feelings when they use the UI created by you?
* Neveloper: Of course not; they do not have such high IQ as I have.
* Nester: Well, you might be better in pure logic than most users, but did you know IQ is only one of the many types of intelligence?
* Neveloper: Sure. It is the only important one.
* Minnie: We are humans. EQ is the only important thing.
* Nester: I believe all of them are important, or at least EQ and IQ, but I consider EQ much more critical.
* Neveloper: That is stupid. Why do you think that?
* Nester: I met people with high IQ who were complete jerks. I will never stay in the same room with such people in the long run.
* Neveloper: And what about the opposite?
* Nester: I do not care what is your IQ if you are a kind and helpful person. Everybody has their own values, it is possible it is not programming, but all of us are good at something. We should focus on that.
* Miki: Neveloper, Nester! I am happy you had such a meaningful discussion, but our readers are here to learn about testing. Can you please continue, Nester?
* Nester: Sure. So here is the updated version of the Humanish Testing Pyramid, AKA The Agileish Humanish Testing Pyramid:

## The Agileish Humanish Test Pyramid



* Miki: Finally, I won't lose my job. Why did you write Automated tests upside down?
* Nester: Well, I messed up something, I defined a 5 minutes timebox, and I couldn't fix it in that timeframe, so I left it like this. Anyway, it is much more memorable this way.
* Miki: Yeah. Sure. Memorable. I would call it ridiculous.
* Nester: Exactly. That is why it is memorable.
* Miki: What types of tests should I write to make these preconditions? I guess I have to create E2E tests to create the preconditions.
* Nester: Nope.
* Miki: Should I write unit tests? That does not make sense.
* Nester: You are right, Miki! Writing unit tests for a tester does not make sense at all. Unit tests are for developers. They are there to provide a safety net for them to refactor and, of course, to show the logic of their program is correct.
* Miki: I guess I have to use SQL and NoSQL to put the data needed for the precondition into the database.
* Nester: Do not even think about that.
* Miki: Why?
* Nester: The database is an implementation detail. Testers have nothing to do with it.
* Miki: I am afraid to ask, but what kind of tests should I write then?
* Nester: You do not have to write automated tests at all.
* Miki: What??? I am fired. No way I can get out of this situation.
* Nester: Cool down, Miki. Cool down. You can keep your job.
* Miki: How?
* Nester: You will automate the process of creating the preconditions.
* Miki: How?
* Nester: By using the app API to create the initial data.
* Miki: I see. Can I at least write a few assertions? What about checking conditions? I will definitely need some there.
* Nester: You should not automate checking postconditions.
* Miki: Why?
* Nester: If something is not visible on the UI, it is irrelevant to the user.
* Miki: Still, I think it would make sense to check some conditions.
* Nester: All right! What kind of postcondition checkings you would like to automate?
* Miki: I don't know. What about asking the readers again?
* Miki, Nester: Dear Reader! What kind of postcondition checks should be automated?
* Miki: So, you would like to downgrade me from a test automation engineer to a test postcondition automation engineer?
* Nester: Nope.
* Miki: Then what?
* Nester: I would like to upgrade you to an automation engineer.
* Miki: Hmmm. I still have to taste this name, but I like it. So what does it mean?
* Nester: Many tasks can and should be automated in a software development company.
* Miki: That is true.
* Nester: So, your job will be to step back, look at the big picture and automate the tasks with which you can get the biggest win.
* Miki: Sounds cool.
* Nester: Yep. It is like in the game Go. You always have to look at the big picture and take the step which brings you the biggest one.
* Miki: What if I start a more significant task and suddenly a new opportunity with an even more substantial value appears?
* Nester: You calculate the remaining effort and win ratio and decide based on that?
* Miki: But then I will have many unfinished tasks. That does not make sense.
* Nester: Well, first of all, the decision is yours. What I said is just a rule of thumb. Secondly, it is scarce that an entirely new opportunity arises which couldn't be seen before.
* Miki: What if it still comes?
* Nester: You will do that and then continue the work with the previous one.
* Miki: What if another new best task for automation comes.
* Nester: In that case, you have to revisit your prioritisation process, as most probably you have made a mistake in evaluating the value of test automation tasks.
* Miki: Umm. Okayish. So, by now, I know what will my job look like in the future. What about humanish manual testers?
* Nester: Well, that is a different story. Let's talk about it.

## The Rise of The Agileish Humanish Tester

* Miki: In the beginning, you said that an average person can become a tester. At least you calculated the salary on average.
* Nester: You are right. Both in terms of the calculation and also in terms of the average person.
* Miki: You will need at least an ISTQB CTFL certification, right?
* Nester: I will never ask for that.
* Miki: Why?
* Nester: Because it does not matter whether you have an ISTQB CTFL or not.
* Mritish Scientistish: I have proved with my fellow Britishish colleagues that you are an incapable tester if you do not have an ISTQB CTFL exam.
* Miki: He might be right. Just look at all the job advertisements on the current job market.
* Nester: It is unimportant.
* Miki: Why?
* Nester: I have to admit, there are valuable things in it, like the 7 testing principle, the software development lifecycles and the testing techniques like equivalence partitioning and boundary value analysis.
* Miki: Yeah, I really loved those parts.
* Nester: But the other half is mostly about standards, which you will never use if you work at an agile company, as they are too cumbersome to use.
* Miki: But there are situations in which you have to use them to prove you are fulfilling the standards needed to release your software in some specific industry area.
* Nester: That is true, but the decision to work at such a place or not is in your hand.

* Miki: So how will the Agileish Humanish Testers learn this info?
* Nester: It is easy; you will write an article about all of them and then attach it to the perfectish matchish partnership opportunity.
* Miki: Ah, you are talking about McRuiter's dream.
* Nester: Exactly.
* Miki: Can you tell me a few words about it?
* Nester: Not right now, but you can learn our previous writing. [How you can find a job in one round of interviews? AKA Perfectish Matchish Partnerish Opportunityish From Developerishish AKA The Great Refactoring](https://mszeles.com/how-you-can-find-a-job-in-one-round-of-interview-aka-perfectish-matchish-partnerish-opportunityish-from-developerishish-aka-the-great-refactoringmd)
* Miki: So you say we should lead our candidates by giving the information based on which they can prepare for the interview?
* Nester: Exactly.
* Miki: But how will you tell whether they just learnt it or already knew it?
* Nester: I do not care. For me, it only matters to get relevant candidates for our company. If somebody comes without knowing the basics despite this, that person is definitely not our perfectish matchish candidate.
* Miki: I heard McRuiter also has one more idea.
* Nester: That guy has so many stupid ideas. What is it this time?
* Miki: He would like to attach the skill matrix describing what knowledge and skillset are needed for junior, medior, senior and lead positions.
* Nester: Wow. But then people could prepare for the interview to make sure they will get hired.
* Miki: That is precisely the type of person I would like to work with.
* Miki: Nester! You would like to find manual testers willing to learn, so I understand the term Humanish Tester. But what does Agileish Humanish Tester mean?
* Nester: It means that this person should be agile.
* Miki: How can they achieve that?
* Nester: Introducing retro(section) into their every day, like you did in the [Agileish super learner article](https://msz.team/3vEc4Mr) and [videos](https://msz.team/3jFifdl).
* Miki: I know what a retro is, but can you please explain to the users who do not know it.
* Nester: Sure. After each iteration, the team gathers together to create 3 lists. During this Scrum ceremony, everybody writes down all the above mentioned three things:
  1. What went well
  2. What went wrong
  3. What will we do to make things better
* Miki: How often should they do this? After every sprint?
* Nester: Yes, after every sprintish.
* Miki: But isn't 2-3 weeks too long to wait for the retro?
* Nester: Who talks about 2-3 weeks? I am talking about 1-2 hours.
* Miki: Whatttt?
* Nester: No matter what you are doing, you can have a retro after each sprintish. The shorter the iteration, the better.
* Miki: Yeah, that is what I am doing in many aspects of my life. Earlier I had a retro after each article. Now I have a retro after each paragraph.
* Nester: Cool. That is what I call agileish.
* Miki: So how should the agileish humanish tester test a feature?
* Nester: He should use a modified version of Edward de Bono's six thinking hats.
* Miki: I  know that method. I have incorporated it into my life. So what is the name of this method?
* MatchyT: Three thinkingish stuffish.
* Miki: Why stuffish?
* Nester: We will talk about that later.
* Miki: What is the relationship between the agileish humanish tester and Edward de Bono's six thinking hats?
* Nester: So, the agileish humanish tester should wear a red hat, black, and retro hat.
* Miki: Ok. So what the testing process will look like?
* Nester: First, the agileish humanish tester should put on the redish stuffish.
* Minnie: That is meeee!
* Miki: Oh, hello, Minnie. I have not seen you for a while.
* Minnie: Yeah, I wanted to let you converse this time.
* Miki: Very excellent of you. So, Nester, can you please explain the meaning of the red hat in the three thinkingish stuff?
* Nester: So when someone is wearing a redish stuffish, he/she/it has to think positively and emotionally.
* Miki: But Edward de Bono has a different hat for positivity. It is the yellow hat.
* Nester: That is true, but we all love to experiment, except Nikolai.
* Nikolai: I do not like anything, especially not if somebody talking about me when I am not here.
* Nester: But, you were here, Nikolai. You are always here.
* Nikolai: I don't care; I hate you anyway.
* Nester: Thanks, Nikolai. You are my best friend.
* Miki: What should I do when wearing the redish stuffish?
* Nester: You should test the positive situations when everything should happen precisely the way it is described in the user story.
* Miki: Ok, so basically, you are testing the happy path. Right?
* Nester: Almostish, Miki! Almostish.
* Miki: What else is?
* Nester: In addition to testing the main functionality, you have to collect all the things which you found good and useful
* Miki: Why?
* Nester: Tell the developer how great the job he/she/it did.
* Miki: Hmm. I have never heard of such processish.
* Nester: Me neither, but I believe it will be valuable.
* Miki: So what is the next one, Nester?
* Nester: You put on a blackish somethingish.
* Miki: Oh! De Bono's black hat!
* Nester: Exactly. By wearing the blackish stuffish, you get the superpower to break every app you touch. It is like Midas' touch, but instead of converting the app to gold, it converts the application to a massive pile of shit.
* Miki: Haha. That was funny.
* Nester: First, the agileish humanish tester should use his creative mind to find as many bugs as possible Then, he/she/it should write down everything terrible about the feature and the application itself.
* Miki: Ok. So what is the third thinkingish stuffish?
* Nester: The third one is the retroish hatch.
* Miki: You mean old school?
* Nester: I mean retrospectives stuffish.
* Miki: Ah, ok. So basically, the agileish humanish tester should have a retro after each sprintish, right?
* Nester: Right.
* Miki: What should he/she/it do during the retroish?
* Nester: First, he/she/it should compile a list of positive things that are good in the application, so based on that, positive feedback can be given to the developer.
* Miki: That makes a lot of sense.
* Nikolai: Feedback is for people with low self-confidence. I do not need feedback.
* Miki: Well, Nikolai, I am sure some feedback could help you step to the next level.
* Nikolai: I am at the top level; I cannot go further.
* Miki: I love your critical thinking, and I like your sarcasm even more. You add a lot to our writings and also to our thinking.
* Nikolai: F\*ck off.
* Miki: What is the problem?
* Nikolai: Some strange Minnieish feeling started to appear in me, so I wanted to make sure it won't take over the control above me.
* Miki: I knew it. Deep inside, all of us have a Minnie; even you have a Minnie, Nikolai.
* Nikolai: We should stop the gibberish talk and continue our conversation.
* Nester: The next step is collecting all the bad things. What went wrong with the tests? What is terrible is the application itself.
* Nikolai: Haha. This is my time. I will tell all the bad stuff to the developers. I want them to feel miserable, and I also expect some good fight, during which they will try to persuade me I am a jerk.
* Nester: You won't tell it to the developers.
* Nikolai: What???? Then what is the point?
* Nester: You will use this list to compile a list of proposed improvements.
* Nikolai: But, that is the same.
* Nester: No, it is not.
* Nikolai: I say it is the same.
* Nester: You are an ugly, useless little piece of sh\*t.
* Nikolai: You are rude, Nester. I like it. I like it very much.
* Nester: It would be awesome if you would be more helpful, and instead of criticising everything brutally, you would give kind feedback by pointing out the areas where I could improve.
* Nikolai: I am helpful. I am always giving my advice to everyone, no matter whether they ask. Regarding being harsh: I am not harsh. I am just honest.
* Nester: However, I can get your feedback without having negative feelings towards you as we have been together for more than three decades; I am sure others might find you offensive. Especially Minnie.
* Nikolai: I would never hurt Minnie's feelings.
* Nester: You already did it a few times. You already did it.
* Nikolai: Nooooo. Really? Sh\*t.
* Miki: Guys! Focus. Focus.
* Nester: As a third step, you put on the retroish stuffish and collect all the potential improvements.
* Miki: I see. That makes sense. What should I do with the list of positive things and improvement suggestions?
* Nester: You have to comment on the ticket/issue that you move from testing to done state.
* Miki: Comment to where?
* Nester: You do have a ticket, right?
* Miki: Sure. Sure. We always have tickets. Alwaysish. So what about the bugs?
* Nester: You are right. You also have to report the bugs in separate tickets and link them to this ticket.
* Miki: Isn't it enough if I just comment my bugs into this ticket?
* Nester: Well, it is up to your team how you do it, but I like to use separate tickets for each bug.
* Miki: Why?
* Nester: Because they are separate bugs, and there is a high chance they will be fixed independently from each other and have a different issue ID, you can have a more readable commit message history.
* Miki: I see. Actually, I am doing that, but I wanted to share our reasonings with the reader.
* Nester: Phew. We are ready. It was a long run.
* Miki: Yeah. That's right. Do you know what is funny, Nester?
* Nester: Of course. It is the platypus.
* Miki: Well, the platypus is really funny, but these animals are even more amusing:
* [embedyt] https://www.youtube.com/watch?v=aZvCwgJgm-Y[/embedyt]
* Nester: Haha. That was a good one. So what is funny?
* Miki: That the agileish humanish tester did not exist until now?
* Nester: What?
* Miki: It was the outcome of my discussion with you guys.
* Nester: Really? So you are not an agileish humanish tester?
* Miki: I am.
* Nester: Since when?
* Miki: From now on.
* Nester: Coool.
* Miki: I think it is time for the call to action.
* Nester: Ok, let's do this together.

Miki, Nester: Dear Reader! Thanks for taking the time to read our article. We hope you found it valuable, even if you hate us now. You can find the original article here:

We are very curious, which one do you like better and why. Please share your thought in the comment section, and do not hesitate to ask if you have any questions!

**BECOME AN AGILEISH SUPERHERO!**

**Learn how to memorise anything in the world with me. In this series, I teach you everything I know about learning and memorisation, including memory palaces, marker images, number memorisation techniques, etc.**

You can already find the first videos on my Youtube channel in which I show you that it is possible to memorise such boring things as positions of circles.

The following online session will be tomorrow, the 17th of April, at 3 am (GMT+1/2, Budapest time). You can ask any question, and I will try to answer them while I try to reach level 103 and more in the Camera Mind memorisation game.

[Just click here to join!](https://msz.team/3NMJ11c)

**My credo**: [The Mystery Of The Supposedly Red ❤ Emoji AKA The Story Of How I Became A Software Developer Detective To Debug The Internet](https://msz.team/36ibgE3)

**Read my story if** you would like **to understand how I think and act as a developer and tester in the real world.** You can **get some insights into my humour**, and I am also happy to tell you that **this is the article about which I am proudest** now.

If you do **not want to miss my other posts**, just **follow me on** [Hashnode](https://hashnode.com/@mszeles), [LinkedIn](https://www.linkedin.com/in/miklos-szeles/), [Twitter](https://twitter.com/mszeles), [Medium.com](https://medium.com/@mszeles), and [dev.to](https://dev.to/mszeles), [Hackernoon](https://hackernoon.com/u/mszeles) and even on [Instagram](https://www.instagram.com/szelesmiki/). 😊

* Minnie: Pssst! Miki!
* Miki: What?
* Minnie: The show is over. Have you seen any exciting stuff in the call to action section?
* Miki: It was just the usual CTA, nothing new.
* Minnie: I asked Mictort to create a painting for us.
* Miki: What?! What it is about?
* Minnie: I asked him to paint the Agileish Humanish Tester.
* Miki: No way! Show it to us!
* Minnie: Here it is: