

THE 7 QUALITIES OF HIGHLY EFFECTIVE BDD SCENARIOS

And How **AI** Can Help You Write Better Ones



FROM BEST-SELLING AUTHOR

John Ferguson Smart

Scenario Outline:

Given Todd has ...

When Todd asks ...

| **Flight Number**

| <Flight Number>

Then the flight ...

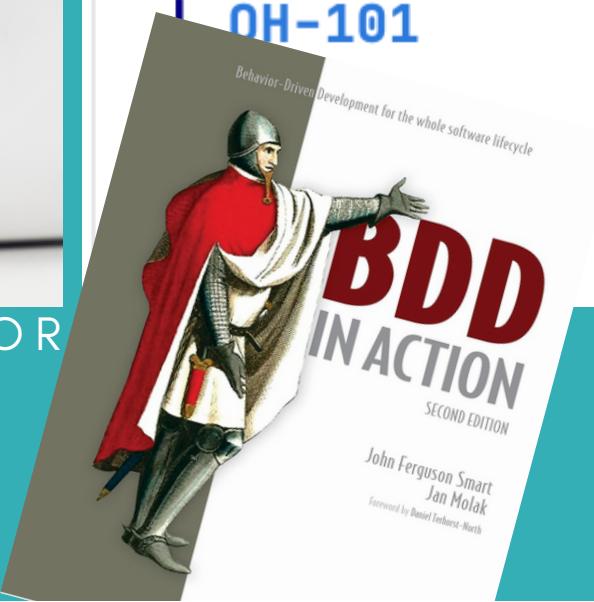
Examples:

| **Flight Number**

| FH-99

| FH-87

| OH-101





HOW TO WRITE HIGHLY EFFECTIVE BDD SCENARIOS WITH EASE

"Writing BDD scenarios is easy!"

That famous "*given..when..then*" notation is so simple and intuitive that anyone can write great BDD scenarios.

Right?

Not so fast!

Sure, great BDD scenarios are easy to read and understand, but making them that way needs a little skill and practice.

And the problem is, most teams **struggle to write high quality scenarios**, or struggle with a test suite **full of poor quality scenarios**.

And all these poorly-written scenarios lead to a **host of problems**, everything from **slow and fragile automated tests**, to **missing features and production defects** and **stressed-out testers** who have no time to complete their normal testing work.

But the good news is, if you can master a few basic tricks, then you can **take your BDD skills to a whole new level**. You'll be able to:

- Write **more precise and more comprehensive scenarios**, catching issues early, way before they lead to embarrassing production defects;
- Write **high quality scenarios faster**, meaning less time in meetings and more time for other activities; and
- Write scenarios that **make automation easier and quicker to implement**, saving you valuable time and effort.

Want to learn more? Let's get going!

Before we start...

Before we start, a gentle word of warning. Maybe you've watched a few YouTube videos, read a tutorial or two, or done a Udemy course on "BDD Test Automation".

If so, be prepared to forget everything you ever learned.

Because highly effective BDD scenarios might be *very* different to what you have seen or learned about....

Read on to learn more...



Tip #1) Write in a Declarative Style

There's a really common mistake tester new to Cucumber and BDD almost always make. They write scenarios as a list of instructions: "click this button", "select that element", "enter this value"; you get the idea.

We call this style of writing "imperative". But the name doesn't matter so much.

What matters is that this style doesn't work so well with Cucumber.

You see, when you write scenarios like this, they get long. They get hard to read. And this makes them *way* too hard for other people to understand and give you feedback on. You end up missing important flows or edge cases, because you are so focused on the minute details of a specific scenario. So you can't use them to make sure you are really building the features you need to be building.

They're also too tied to the implementation, so if the implementation changes, the scenario breaks, even if the business rule is still valid.

Instead, write in a **declarative** style.

Declarative scenarios focus on **describing the problem to solve** and the **business rules to implement**. They illustrate these rules with **clear, readable examples** written in the language of the customer.

```

Scenario: Approving KYC review
  Given the user logs in as: user@email.com
  And the user navigates to My Reviews
  When the user selects a review category: 'KYC'
  And the user selects the first review
  And the user clicks on 'Open Review'
  Then the Review Details screen should be opened
  When the user fills in the comments field with 'All good'
  And the 'Save' button is clicked
  Then the validation message is shown
  When the user clicks on 'OK'
  Then show the review list

```



```

Scenario Outline: Customers in high risk business must be escalated to a senior analyst
  Given Joe has been assigned the following review:
    | customer | business risk rating | status | assigned to |
    | <customer> | <risk> | <status> | <assigned to> |
  When Joe approves the review
  Then the review should be updated to:
    | status | assigned to |
    | <final status> | <final assigned to> |
Examples:
    | customer | risk | status | assigned to | final status | final assigned to |
    | Alistair Capone | high | pending | Joe | pending approval | Unassigned |
    | Bonnie Clvde | medium | pending | Joe | approved | Senior Analvst |

```

Tip #2) Use in Business Language

This one is *really* important.

Gherkin is above all about **communication** - communication within the team, and with the business or users.

So make sure the text reads smoothly, even to a non-tester. Make sure it speaks to the business person, and uses the words they would use. Check that there aren't any grammatical errors that distract the reader. **Be kind to your reader.**

Tip #3) Don't Write Test Scripts in Cucumber

Some folk mistake Cucumber for a testing tool.

And they write Cucumber scenarios that read like traditional test scripts, as if they were telling another manual tester how to test the feature.

This doesn't work well in Cucumber. You'll end up with a mess of **brittle test scripts** that will **slow you down** and **cost heaps to maintain**.

So use Cucumber to describe and discuss your business rules and key flow and outcomes, not to write test scripts.

Scenario: Escalating a KYC review

```
Given the user logs in as: some.user@email.com
And the user navigates to My Reviews
When the user opens the first review
And the user enters the comment Dodgy character
And the user clicks on 'Escalate'
Then check that the review list is updated correctly
```

Scenario: Escalating a KYC review

```
If a level 1 analyst escalates a review, it goes to the level 2 analyst list
Given the following reviews
| customer | status | assigned to |
| Alistair Capone | pending | Sam |
| Bonnie Clyde | pending | Sam |
And that Sam has opened his KYC reviews list
When Sam escalates the review for Bonnie Clyde with the comment 'Dodgy character'
Then Sam's review list should contain:
| customer | status | assigned to |
| Alistair Capone | pending | Sam |
And Mandy's review list should contain:
| customer | status | assigned to |
| Bonnie Clyde | escalated | Mandy |
```



Tip #4) Be Concise

A well written scenario is short and sweet.

It doesn't list off reams of detailed steps; it doesn't try to give you a blow-by-blow description of what the user is doing.

Instead, it focuses on the key business tasks the user wants to perform, and the outcomes they want to achieve.

The details, the clicks and selects, the API calls, the database queries: all of that should be hidden away, out of sight of the reader.

And generally half a dozen steps or less is plenty enough.

Tip #5) Identify Tangible Outcomes

A good scenario doesn't leave you hanging.

It describes the **concrete business outcome** the business user wants.

Some scenarios try to specify what fields or buttons on a page, like this one here:

```
Scenario: Users can filter articles by status
  Given the user opens the 'Pending Review' article page
  When the user clicks on the "Technology" tab
  Then the user should see the Status Filter dropdown
  And the Status Filter dropdown should contain:
    | Overdue   |
    | Due today |
    | Upcoming  |
```

But Gherkin is not a great way to try to describe a user interface - it quickly becomes cumbersome, and it's not a format users will be happy to give feedback on anyway.

Good scenarios don't simply check whether certain fields or elements appear on a page; they show how these fields or elements are used to achieve some useful outcome.

Tip #6) Be Informative, Not Overwhelming

Good scenarios provide just enough information to explain the behaviour, but not too much to make it hard to understand. They avoid including unnecessary additional data.

Because just as scenarios that are too long are hard to read, scenarios with too much data are also hard to read.

Scenario: Publish new articles on social media

Given today is **05/08/2021**

And the following articles are awaiting review

id	title	priority	category id	category name	due date	subcategory	created	author
12321342	Etherum rising	medium	123	Technology	03/08/2021	Cybercurrencies	01/08/2021	Sarah
23534635	Climate Change News	medium	124	Environment	05/08/2021	Climate Change	01/07/2021	Jill
623423	Cat pics still cool	medium	123	Technology	07/08/2021	Cool stuff	01/08/2021	Paul
2342534	Big on Bitcoin	medium	123	Technology	17/08/2021	Cybercurrencies	12/07/2021	Peter

When Ted approves the articles: **12321342, 23534635**

Then the following Social Media tasks should be generated:

taskid	articled	title	platform	status code	queued	tags
12312	12321342	Etherum rising	Twitter	P	3	#Technology #Cybercurrencies
12313	12321342	Etherum rising	Facebook	P	3	#Technology #Cybercurrencies
12314	12321342	Etherum rising	LinkedIn	P	3	#Technology #Cybercurrencies
12315	23534635	Climate Change News	Twitter	P	3	#Environment #ClimateChange
12316	23534635	Climate Change News	Facebook	P	3	#Environment #ClimateChange
12316	23534635	Climate Change News	Facebook	P	3	#Environment #ClimateChange

Scenario: Publish new articles on social media

And the following articles are awaiting review

title	category	subcategory
Etherum rising	Technology	Cybercurrencies
Climate Change News	Environment	Climate Change
Cat pics still cool	Technology	Cool stuff
Big on Bitcoin	Technology	Cybercurrencies

When Ted approves the articles: **"Etherum rising", "Climate Change News"**

Then the following Social Media tasks should be generated:

platform	title	tags
Twitter	Etherum rising	#Technology #Cybercurrencies
Facebook	Etherum rising	#Technology #Cybercurrencies
LinkedIn	Etherum rising	#Technology #Cybercurrencies
Twitter	Climate Change News	#Environment #ClimateChange
Facebook	Climate Change News	#Environment #ClimateChange
Facebook	Climate Change News	#Environment #ClimateChange



Tip #7) Start From The Outcomes

In real-world applications, the real value comes from the outputs or outcomes, not from the inputs. For example, a search field is of little value if the search button doesn't work, so a scenario that simply checks for the presence of the search field is not very useful.

Given I'm on the homepage

When I click on "Search"

Then I should see the search form



We also need to be able to verify these outcomes. So a good scenario is also easy to verify - the outcome is clear and unambiguous, and can be checked automatically, without a tester having to intervene.

Then the *right* values are displayed



Then the trade is processed *quickly*



The Good Gherkin Score Card

You can use this score card to evaluate the approximate quality of your Cucumber scenarios, and (more importantly) to avoid some of the pitfalls of poor quality ones.

Score	Interpretation
4 or under	Very poor
5-6	Poor
7	Showing signs of improvement
8	Getting better
9-10	Good

Category	Question	Weight	Yes/No
<i>Declarative not imperative</i>	The scenario describes an outcome to achieve, not how to achieve that outcome	1 point	
<i>Business Language</i>	The scenario is grammatically correct	1 point	
<i>Business Language</i>	The scenario is easy to understand for a business person	1 point	
<i>Business Language</i>	The scenario does not use implementation details	1 point	
<i>Not a test script</i>	The scenario describes how the system behaves, not what the tester should do	1 point	
<i>Concise, not convoluted</i>	The scenario has no more than 5 steps	1 point	
<i>Tangible Outcomes</i>	The scenario has clearly defined inputs and outcomes expressed in business terms	1 point	
<i>Informative, not overwhelming</i>	All data in the scenario is required to understand the scenario	1 point	
<i>Specific and Actionable</i>	Inputs are clear, well-defined and self-evident	1 point	
<i>Specific and Actionable</i>	Outputs are verifiable and can be related back to the inputs	1 point	

BUT HOW CAN AI HELP?



Generative AI took the world by storm in 2023, raising a critical question among many: **Can AI write my BDD scenarios for me?**

The answer is...it's complicated.

AI is undeniably **a game-changer for learning and producing higher-quality BDD scenarios** more quickly... BUT the majority of data that tools like ChatGPT have been trained on is of very poor quality or just downright wrong.

This means that the Gherkin scenarios it generates right off the bat... well, let's just say your mileage may vary. **ChatGPT can and will mislead you if you're not careful.**

In this section, we'll explore practical ways you can harness AI effectively to not only learn how to **craft better BDD scenarios** but also to **speed up the writing process**.

1) Leverage The Power Of Structured Prompts

The quality of the BDD scenarios that ChatGPT produces by default can vary a lot. But you can increase your chances of getting well written outputs by telling ChatGPT what you want upfront. You do this using an initial prompt, before you start asking for scenarios.

This one works well:

I want you to help me write high quality BDD scenarios for a business requirement that I will give you. The scenarios should be written in a declarative style. They should use concrete examples rather than general statements wherever this will make the business logic clearer and easier to verify. They should focus on business needs and not commit to specific implementation details, such as buttons or fields. They should use data tables rather than repetitive sentences Do you understand?

J You

I want you to help me write high quality BDD scenarios for a business requirement that I will give you. The scenarios should be written in a declarative style. They should use concrete examples rather than general statements wherever this will make the business logic clearer and easier to verify. They should focus on business needs and not commit to specific implementation details, such as buttons or fields. They should use data tables rather than repetitive sentences Do you understand?

ChatGPT

Yes, I understand.



J You

Can you help me write BDD scenarios for this user story:

As an online banking client, I want to transfer funds from my current account to my savings account so that I can save more money.

ChatGPT

Certainly! Writing BDD (Behavior-Driven Development) scenarios for this user story involves creating clear, detailed, and example-driven specifications. These scenarios should describe the desired behavior of the system without focusing on the implementation details. Here's how they might be structured:

Feature: Fund Transfer in Online Banking

User Story: As an online banking client, I want to transfer funds from my current account to my savings account so that I can save more money.

Scenario Outline: Successful transfer of funds from current to savings account

- Given the client has a current account and a savings account
- And the current account has a balance of `<current_account_balance>`
- When the client initiates a transfer of `<transfer_amount>` to the savings account
- Then the transfer should be successful
- And the new current account balance should be `<new_current_account_balance>`
- And the new savings account balance should be `<new_savings_account_balance>`

current_account_balance	transfer_amount	new_current_account_balance	new_savings_account_balance
5000	500	4500	1500
2000	1500	500	2500
1000	100	900	1100

Scenario: Transfer fails due to invalid amount

- Given the client has a current account and a savings account
- When the client attempts to transfer a negative amount
- Then the transfer should fail
- And an error message 'Invalid transfer amount' should be displayed

2) Iterate on the answers

Even with structured prompts, ChatGPT does not produce flawless Gherkin. ChatGPT can make things go faster, but it is always important for you to know what good Gherkin looks like. Some of the scenarios it produces might also be incorrect or inapplicable.

Once ChatGPT has come up with an initial response, give it your feedback and ask for a more refined version. For example, some of the things you might ask are:

- *“Can you use data tables to make the examples more concise”*
- *“Can you write these as well-formed Gherkin features in code snippets”*
- *“Can you rewrite this in more business-focused language without referring to the user interface or UI interactions”*

 **You**
Can you write these as well-formed Gherkin features in code snippets.

 **ChatGPT**
Absolutely, I can format the BDD scenarios into well-formed Gherkin syntax for you. Gherkin is a language used to define tests in Behavior-Driven Development (BDD) and is known for its readable, plain-text description of software features. Here's how your scenarios would look in Gherkin:

```
gherkin Copy code
Feature: Fund Transfer in Online Banking

User Story: As an online banking client, I want to transfer funds from my current account to my savings account so that I can earn interest.

Scenario Outline: Successful transfer of funds from current to savings account
  Given the client has a current account and a savings account
  And the current account has a balance of <current_account_balance>
  When the client initiates a transfer of <transfer_amount> to the savings account
  Then the transfer should be successful
  And the new current account balance should be <new_current_account_balance>
  And the new savings account balance should be <new_savings_account_balance>

Examples:
| current_account_balance | transfer_amount | new_current_account_balance | new_savings_account_balance |
| 5000                   | 500           | 4500                         | 1500                        |
| 2000                   | 1500          | 500                          | 2500                        |
| 1000                   | 100           | 900                          | 1100                        |

Scenario Outline: Transfer fails due to insufficient funds
  Given the client has a current account and a savings account
  And the current account has a balance of <current_account_balance>
  When the client attempts to transfer <transfer_amount> to the savings account
  Then the transfer should fail
  And an error message 'Insufficient funds' should be displayed
```

3) Give ChatGPT As Much Context As Possible

The more context and background you can give to ChatGPT, the better the results will be. And here's a little trick: by writing down your ideas and assumptions, it helps you get more clarity about the requirements. So don't just give ChatGPT a simple user story - flesh it out with details. Give a bullet-pointed list of the rules and acceptance criteria you can think of. Give some examples. It doesn't have to be well-written, or even perfect English, but do make it as detailed and complete as you can - any information you can give will help.

 **You**

Given this user story and its basic acceptance criteria, can you expand them into a detailed Gherkin scenario, considering edge cases and typical user behavior?

Here is the user story:

As an online banking client
 I want to be able to have different savings accounts with different names
 So that I can organise my savings more effectively
 - Account names can be no longer than 30 characters
 - Account names must be unique
 - Account names can have emojis or special characters in them
 - An account holder can have up to 20 different named savings accounts
 - The account holder should be able to easily see the total amount saved, and the amount in each named account

 **ChatGPT**

Certainly! Expanding the provided user story and its acceptance criteria into a detailed Gherkin scenario involves considering various edge cases and typical user behaviors. Here's a set of Gherkin scenarios that cover the requirements:

```
gherkin
Copy code

Feature: Managing Multiple Savings Accounts

User Story: As an online banking client, I want to be able to have different saving

# Scenario for creating a new savings account with valid name
Scenario Outline: Successfully creating a new savings account with valid name
  Given the client is logged into their online banking account
  When the client attempts to create a new savings account with the name "<account_
  Then the savings account should be successfully created with the name "<account_r

Examples:
| account_name      |
| Vacation Fund    |
| Emergency 💰     |
| New Car 🚗 Savings |
| Birthday 🎉 Fund  |

# Scenario for account name uniqueness
Scenario: Attempting to create a savings account with a duplicate name
  Given the client has a savings account named "Holiday Fund"
  When the client attempts to create another savings account named "Holiday Fund"
  Then the account creation should fail
  And an error message 'Account name must be unique' should be displayed

# Scenario for account name length
Scenario: Attempting to create a savings account with a name exceeding 30 character
```



Next Steps? Try Out The New BDD Guru GPT

Writing BDD scenarios can often be a challenge, particularly for those new to it. Out-of-the-box, ChatGPT might not provide the assistance you need without significant tweaking and an understanding of effective Gherkin.

That's why I've created BDD Guru, a brand-new custom-trained AI tool designed to support your BDD practices. It's been trained on quality BDD and Gherkin examples to help you improve your scenarios. Whether you're looking to refine what you have or need fresh scenarios for your stories, BDD Guru GPT is ready to help.

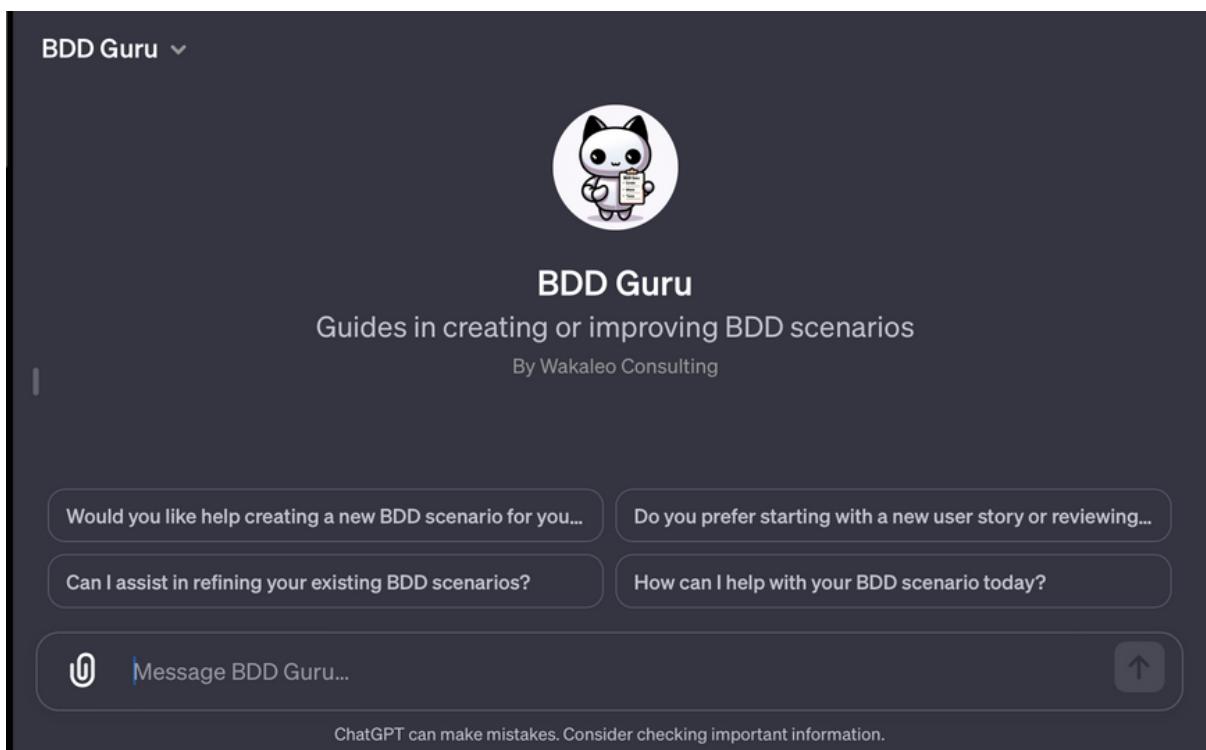


I've personally trained BDD Guru using my own personal documents, training material and hundreds of examples of good quality real-world scenarios to save you time and effort when it comes to generating high quality Gherkin.

Whether you're refining existing scenarios or brainstorming new ones, BDD Guru GPT is like having an expert by your side, offering suggestions that consider the complexities and subtleties of your user stories. See the difference it makes when AI is aligned with BDD excellence.

Ready to revolutionize your scenario writing process?

Dive into BDD Guru GPT now and see the impact for yourself!



BDD Guru ▾



BDD Guru
Guides in creating or improving BDD scenarios
By Wakaleo Consulting

Would you like help creating a new BDD scenario for you...
Do you prefer starting with a new user story or reviewing...
Can I assist in refining your existing BDD scenarios?
How can I help with your BDD scenario today?

 Message BDD Guru... 

ChatGPT can make mistakes. Consider checking important information.

Conclusion

AI can certainly help you write better BDD scenarios faster, but knowing what good Gherkin looks like is still essential. AI supports and enhances communication and scenario writing, but it cannot replace collaboration or conversations that are at the heart of the BDD process. The goal of BDD is to build a common understanding so you can deliver valuable features faster, and Gherkin is a language we use to make sure everyone is on the same page.

In other words, use AI to streamline the scenario writing task, transforming it from a chore into a more manageable and perhaps even enjoyable part of your development workflow.

SO WHAT COMES NEXT?

Whether you're looking to improve your skills in test automation, or build a high-performing agile team, BDD is a real game changer. And, as we all know, a little help can go a long way!

So what are your goals?

"I want to master BDD and take my test automation skills to the next level"



Looking for a boost in your agile test automation career? We've got your back!

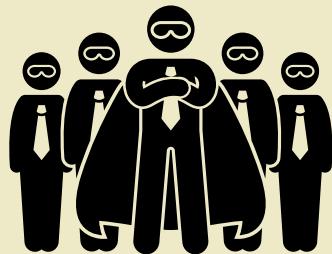
At the Serenity Dojo we have helped hundreds of testers from all background become **high performing agile test automation engineers**, with a solid understanding both of the automation tools, and of the best way to use the tools in modern projects.

Our approach blends **innovative training, personalized coaching, interactive live sessions, and ongoing support**, all designed to elevate your proficiency to its peak. Join us, and **unlock your potential to become the best in the field**.



[LEARN MORE](#)

"I want to help my team adopt BDD so that they can deliver higher quality features that delight our customers"



Transform your team's capabilities with our BDD Accelerator workshop - a **lean, cost-efficient path** to mastering Behaviour Driven Development for the whole team.

Perfected over a decade, our unique program distills practical, powerful lessons into an engaging, interactive experience tailored to your domain that's unlike any other.

We've guided hundreds of teams to excel in BDD, streamlining their processes and supercharging their delivery of high-quality software.

Ready to lead your team to the pinnacle of BDD proficiency? Discover the BDD Accelerator difference.



[LEARN MORE](#)