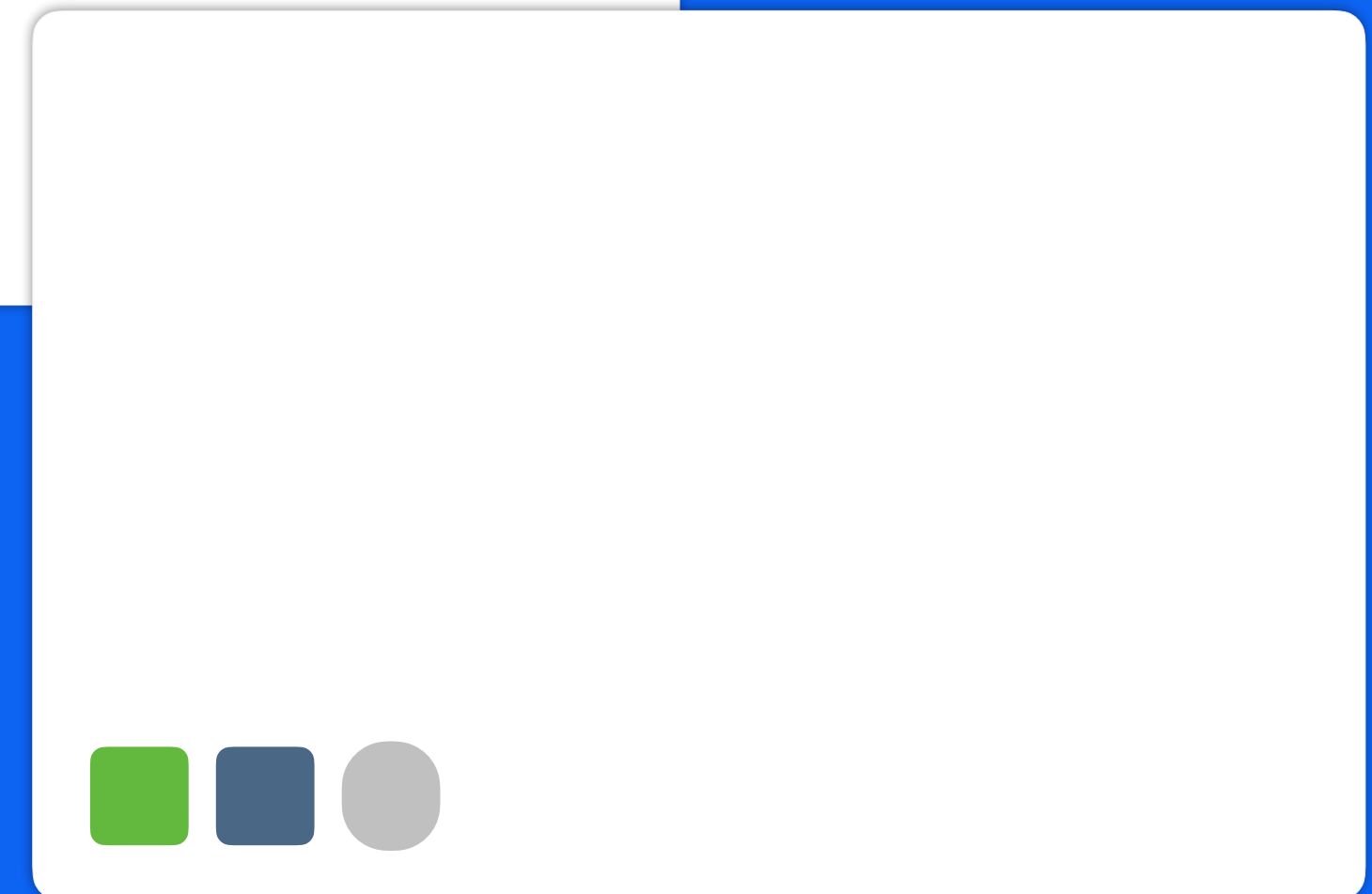


WORKSHOP

---

# Writing User Stories



# contents

SECTION

**1**

## **user stories**

what is a user story?	4
user story template	5
examples: user stories	6
user story checklist	7
why not tasks?	8

SECTION

**2**

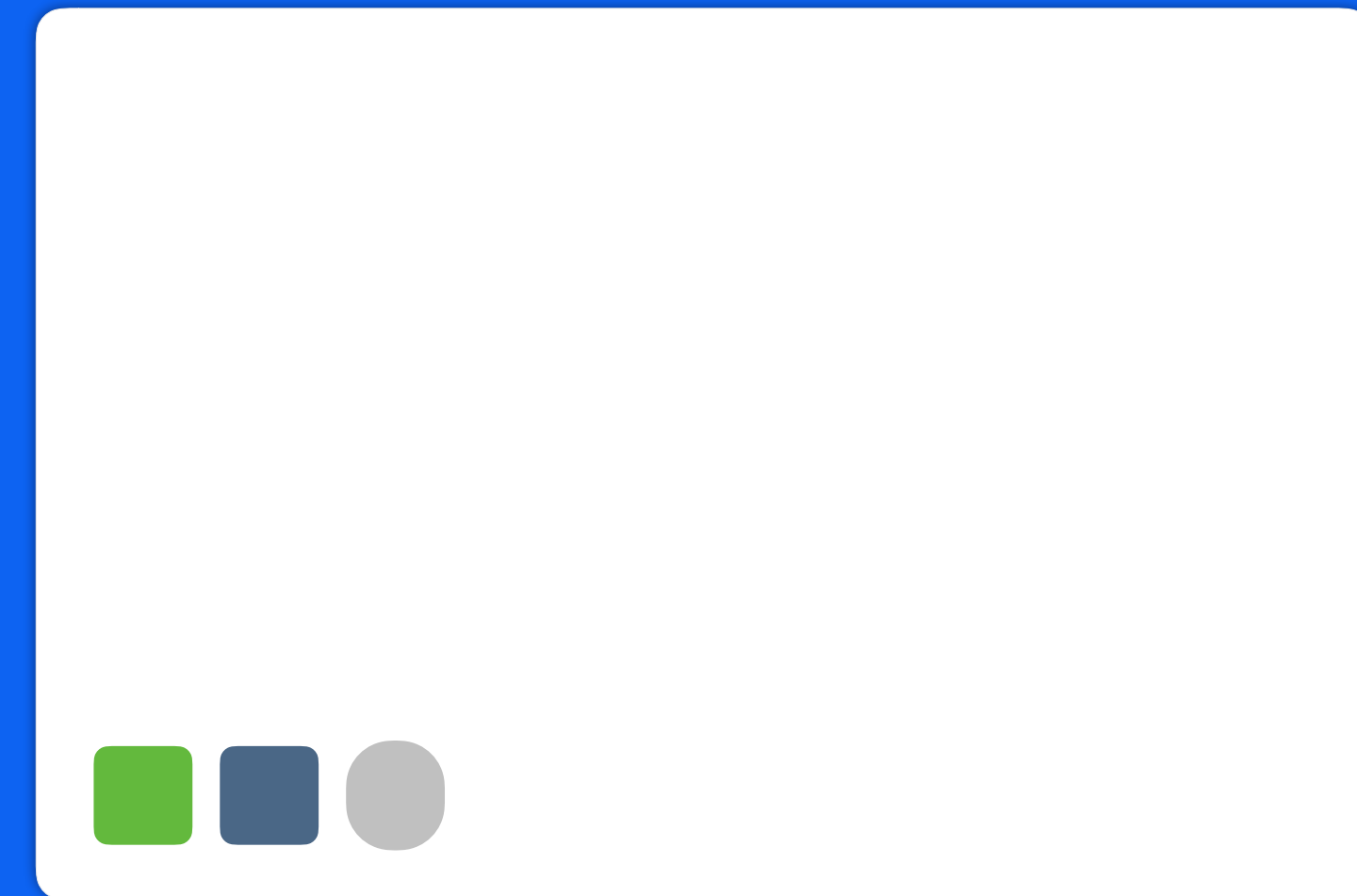
## **writing acceptance criteria**

what is acceptance criteria?	10
example: acceptance criteria	11
acceptance criteria checklist	13

WORKSHOP

---

# What Is A User Story?



# definition: user story

*A user story is a tool used in agile software development to capture the description of a software feature from an end-user perspective. The user story describes the type of user, what they want and why, A user story helps to create a simplified description of a requirement.*

A user story often follows the following 'equation':

**As a <type of user>, I want <some feature> so that <reason>**

A simple example of this could be:

**As an online shopper, I want to add an item to my cart, so that I can purchase it**

# user story template

<b>WHO</b> are we building it for? Who is the user?	As a <type of user>
<b>WHAT</b> are we building? What is the intention?	I want <some goal or objective>
<b>WHY</b> are we building it? What is the value for the customer?	So that <benefit/value>

# examples: user stories

**As an** internet banking customer

**I want** to see a rolling balance for my everyday accounts

**So that** I know the balance of my account after each transaction is applied



**As an** administrator

**I want** create other administrators

**So that** I can delegate tasks



**As a** marketer

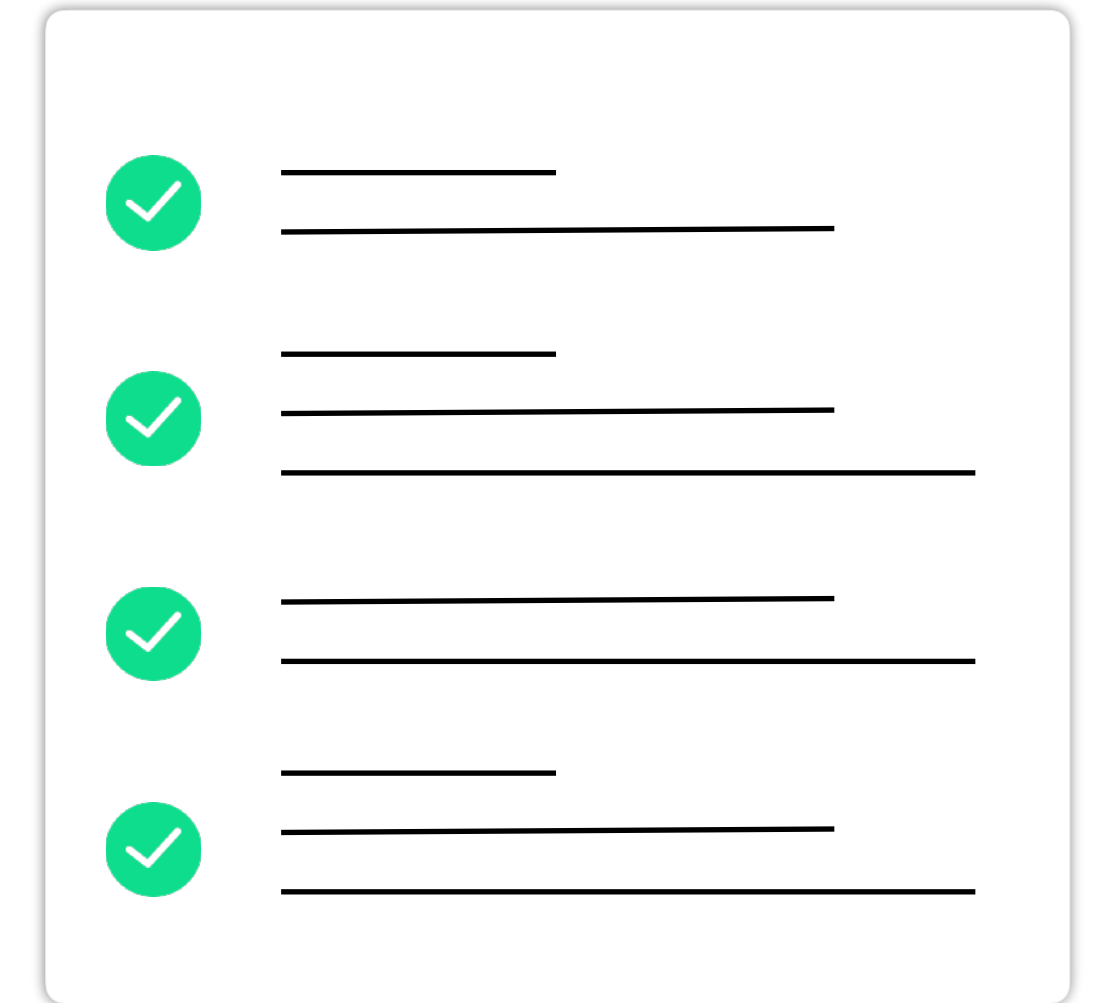
**I want** create automated email campaigns

**So that** I can keep evaluators engaged



# user story checklist

- Keep them short
- Keep them simple
- Write from the perspective of the user
- Make the value/benefit of the story clear - what is the reason for the story?
- Describe **one** piece of functionality. If you have to write **and** break it into 2 stories
- Write stories as a team
- Use acceptance criteria to show a MVP



- \_\_\_\_\_
- \_\_\_\_\_
- \_\_\_\_\_
- \_\_\_\_\_

# why not just use 'tasks'?



## user stories

a user story = the WHAT

user stories describe a piece of functionality from the point of view of the user

divided features into business processes



## tasks

the task = the HOW

"what are the activities we need to perform in order to deliver outcomes (user stories)"

tasks are individual pieces of work



WORKSHOP

---

# Writing Acceptance Criteria

# definition: acceptance criteria

*Acceptance criteria or 'conditions of satisfaction' provide a detailed scope of a user's requirements. They help the team to understand the value of the story and set expectations as to when a team should consider something done.*

## Acceptance Criteria Goals:

- to clarify what the team should build before they start work
- To ensure everyone has a common understanding of the problem
- To help the team members know when the story is complete
- To help verify the story via automated tests

# example: acceptance criteria

*As an online banking customer, I want strong a strong password, so that my credit card information is secure*

## Acceptance Criteria:

- The password must be at least 8 characters
- The password must contain at least 1 character from each of the following groups: lower case alphabet, upper case alphabet, numeric, special characters (!, @, #, \$, %, ^, &, \*)

# example: acceptance criteria

*As a conference attendee, I want to be able to register online, so that registration is simple and paperless*

## Acceptance Criteria:





- A user can not submit a form without filling out all of the mandatory fields
- Information from the form is stored in the registrations database
- Protection against spam is working
- Payment can be made via Paypal, Debit and Credit Card
- An acknowledgment email is sent to the attendee after submitting the form

# acceptance criteria should include

- Negative scenarios of the functionality
- Functional and non-functional use cases
- Performance concerns and guidelines
- What system/feature intends to do
- End-to-user flow
- The impact of a user story to other features
- UX concerns



# acceptance criteria should NOT include

-  Code review was done
-  Non-blocker or major issues
-  Performance testing performed
-  Acceptance and functional testing done

## why?

*Your acceptance criteria should not include any of the above, because your team should already have a clear understanding of what your Definition of Done (DoD) means. This could mean:*

- unit/integrated tested*
- ready for acceptance test*
- deployed on demo server*
- releasable*

