Coding Bootcamp

Computational Thinking

Module 01

© 2024 edX Boot Camps LLC. Confidential and Proprietary. All Rights Reserved.

What is computational thinking?

Computational thinking is a way of logically breaking down a problem so that we can develop a step-by-step solution.

Computational Thinking

Key principles of computational thinking include the following:

1. Decomposition
2. Pattern recognition
3. Abstraction
4. Algorithms

What is decomposition?

Decomposition

We use **decomposition** to break down a problem down into smaller, more manageable parts.

What is pattern recognition?

Pattern recognition

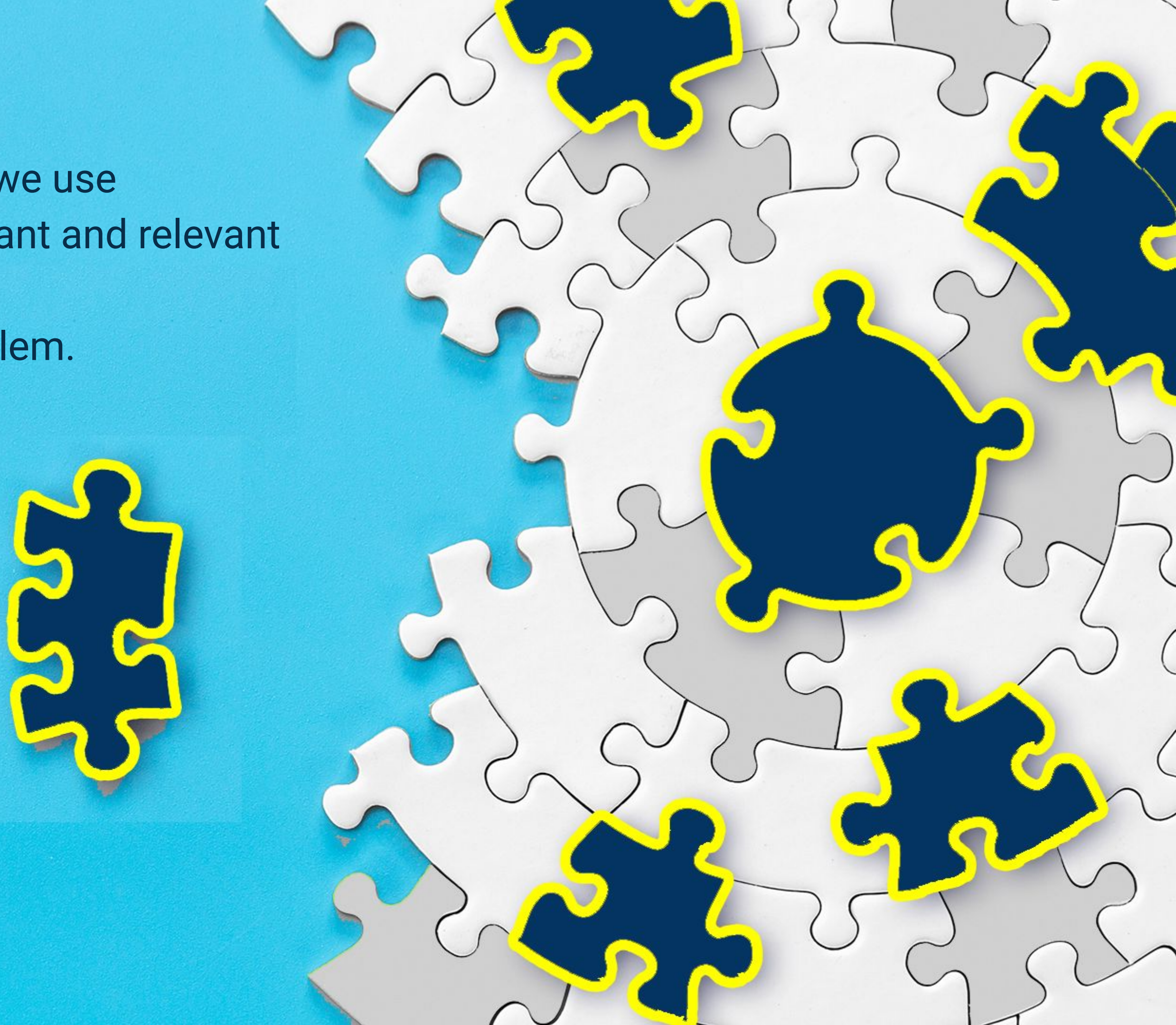
Once a problem is broken down, we use **pattern recognition** to find similarities and patterns among the smaller parts.

This helps us solve the problem more efficiently.

What is abstraction?

Abstraction

Once patterns are recognized, we use **abstraction** to focus on important and relevant information and filter out what is not needed to solve our problem.



What is an algorithm?

After the other steps are complete, it is time for a plan. An **algorithm** is a set of step-by-step instructions that provide a solution to the problem.

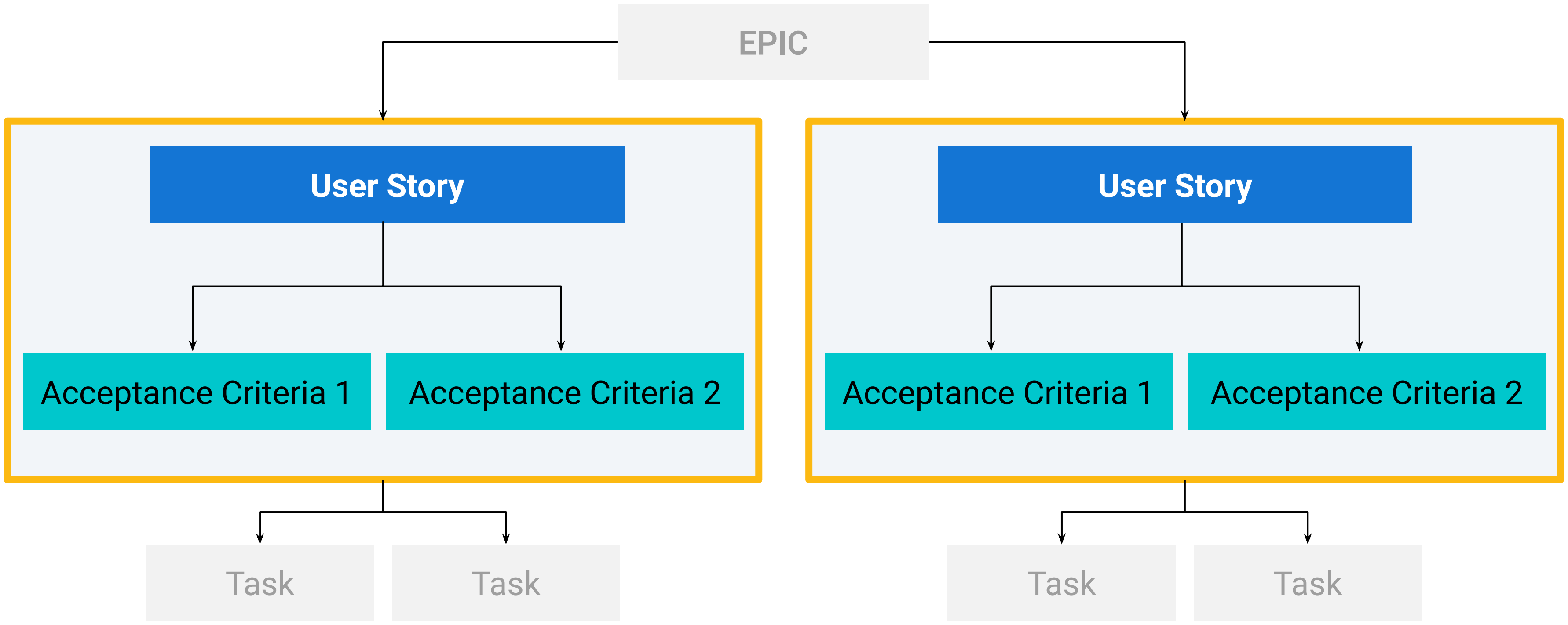
How to Solve a Puzzle

1. Choose a puzzle.
2. Flip all pieces face up.
3. Find all the edge pieces.
4. Use edge pieces to create frame.
5. Group the remaining pieces by color.
6. In each color group, find the special pieces.
7. Break down the image into small sections.
8. Work on one section at a time, paying attention to colors and special pieces.
9. Join completed section inside of frame.

How do we apply computational thinking to solve coding challenges?

Our Problem-Solving Tools of the Trade

User stories and acceptance criteria help us clarify the scope of the problem and break it down into manageable parts.



A **user story** helps us identify the end goal and communicate how the solution will meet that goal.

Identifying Goals and Rewards: User Stories

A user story follows a specific format:

As a **[user]**,

I want to **[do something]**,

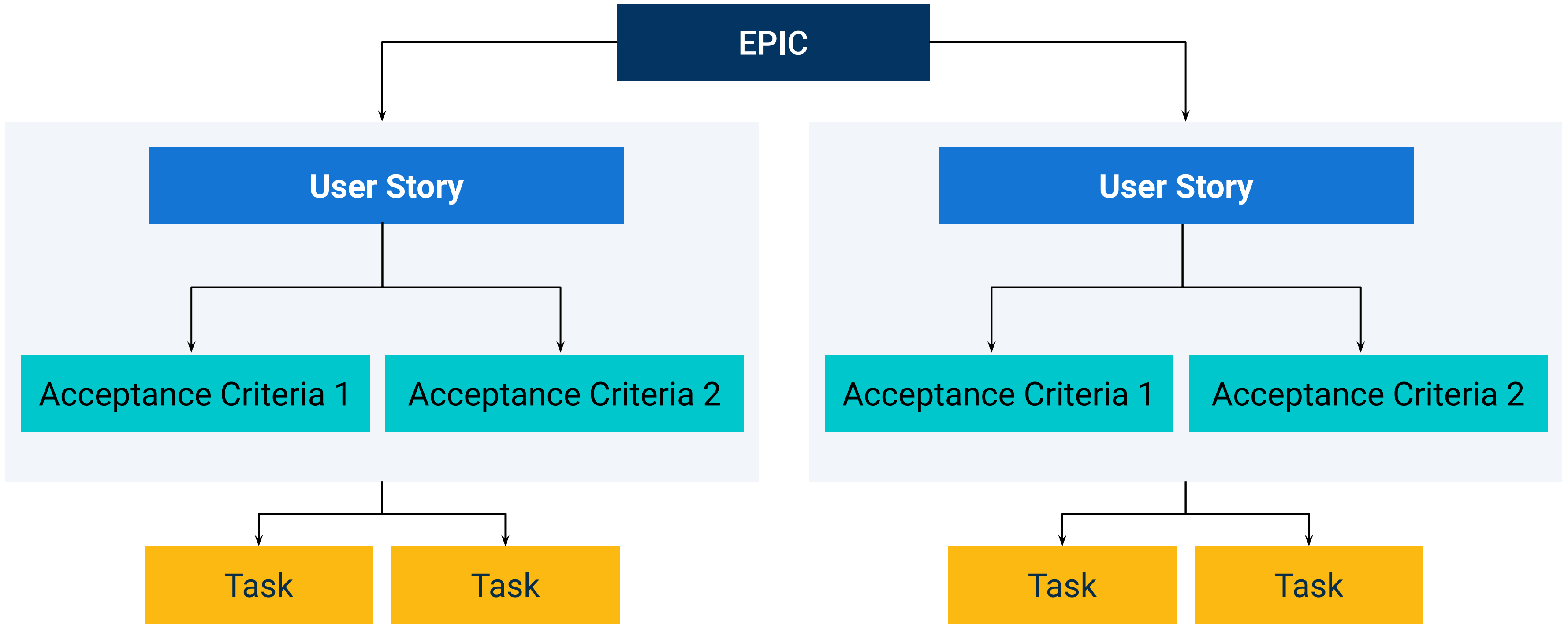
so that I can **[realize a reward]**.

[user]	The end user or customer.
[do something]	Identifies the goal the solution must address.
[realize a reward]	Describes when the goal is met.

Acceptance criteria provide the conditions that must be met for the goal to be accomplished and the solution accepted by the user.

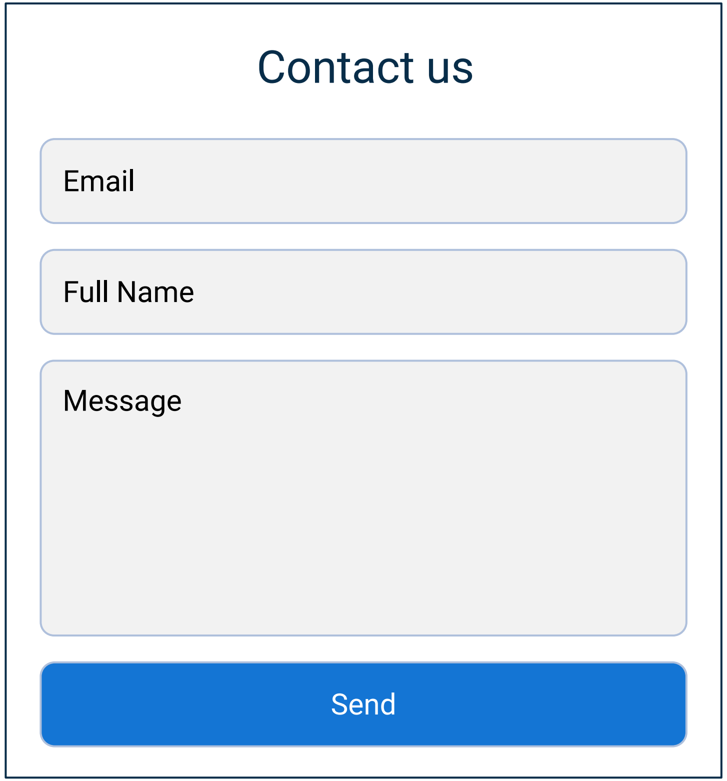
Getting Specific: Acceptance Criteria

Acceptance criteria define what specific tasks or functions must be done to solve the problem presented in the user story and have a clear pass or fail result. All criteria must pass for the solution to be accepted.



Getting Specific: Acceptance Criteria

For example: It's done when there is a contact form that includes a text box for a visitor's name and email.



The solution is a set of **step-by-step instructions** that address each of the acceptance criteria and clearly **meet the user's goal** and **reward as described in the user story.**

Instructor **Demonstration**

Mini-Project

Questions?

The End