GOTO
Guide

LET US HELP YOU

Ask questions through **the app**

★★★★★

also remember to rate session

THANK YOU!

#GOTOams

Why Should You Look into Low Code?

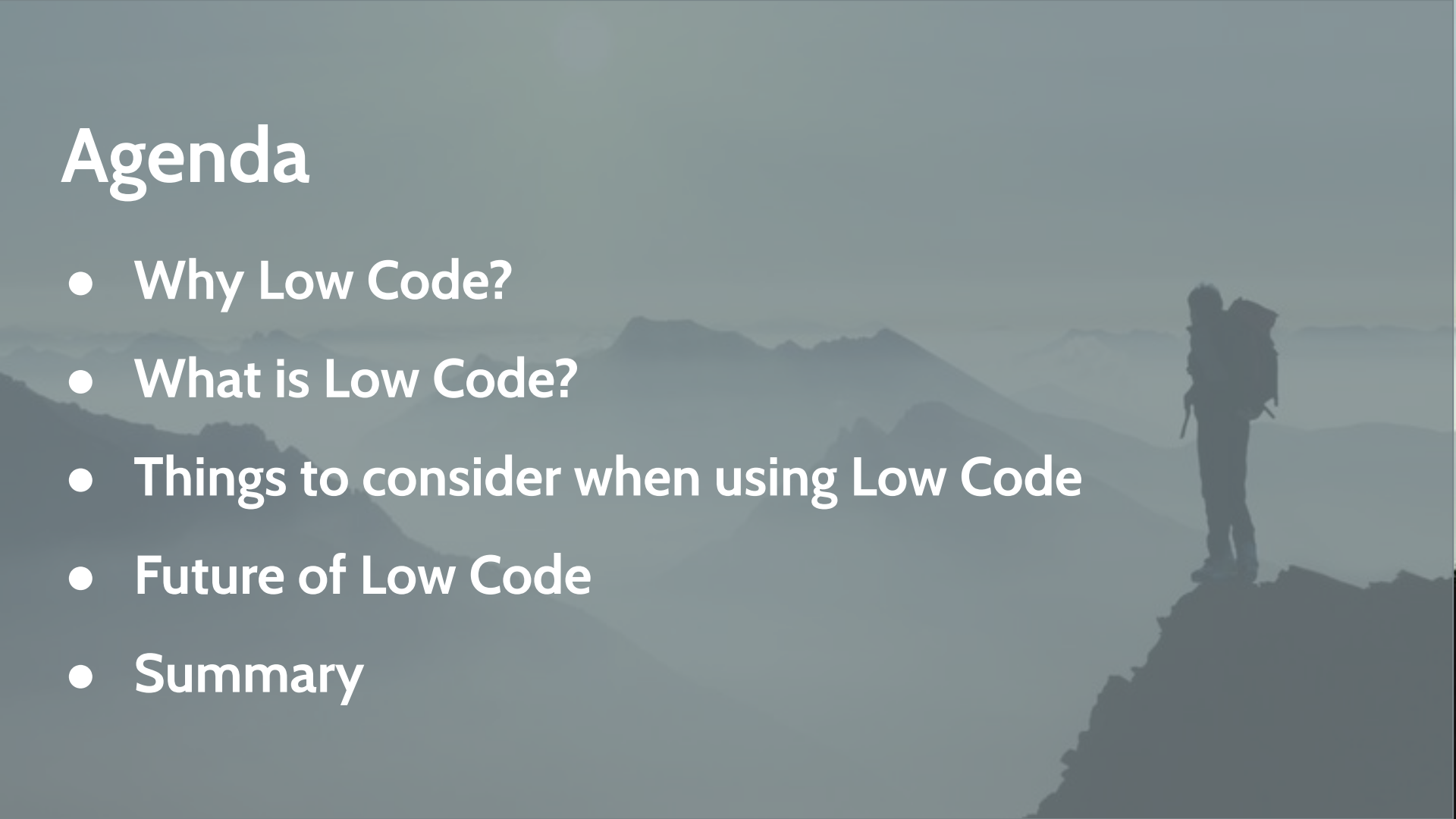
Christoph Windheuser, OutSystems
GOTO Amsterdam
June 15, 2022

“A lot of low-code tools are being promoted with this kind of naive starting point that code is somehow bad, and that rankles a lot with programmers,”

Mike Mason, Global Head of Technology at Thoughtworks
in *"Does Low Code Mean More Work or More Freedom for Developers?"*,
THENEWSTACK, Feb. 2022

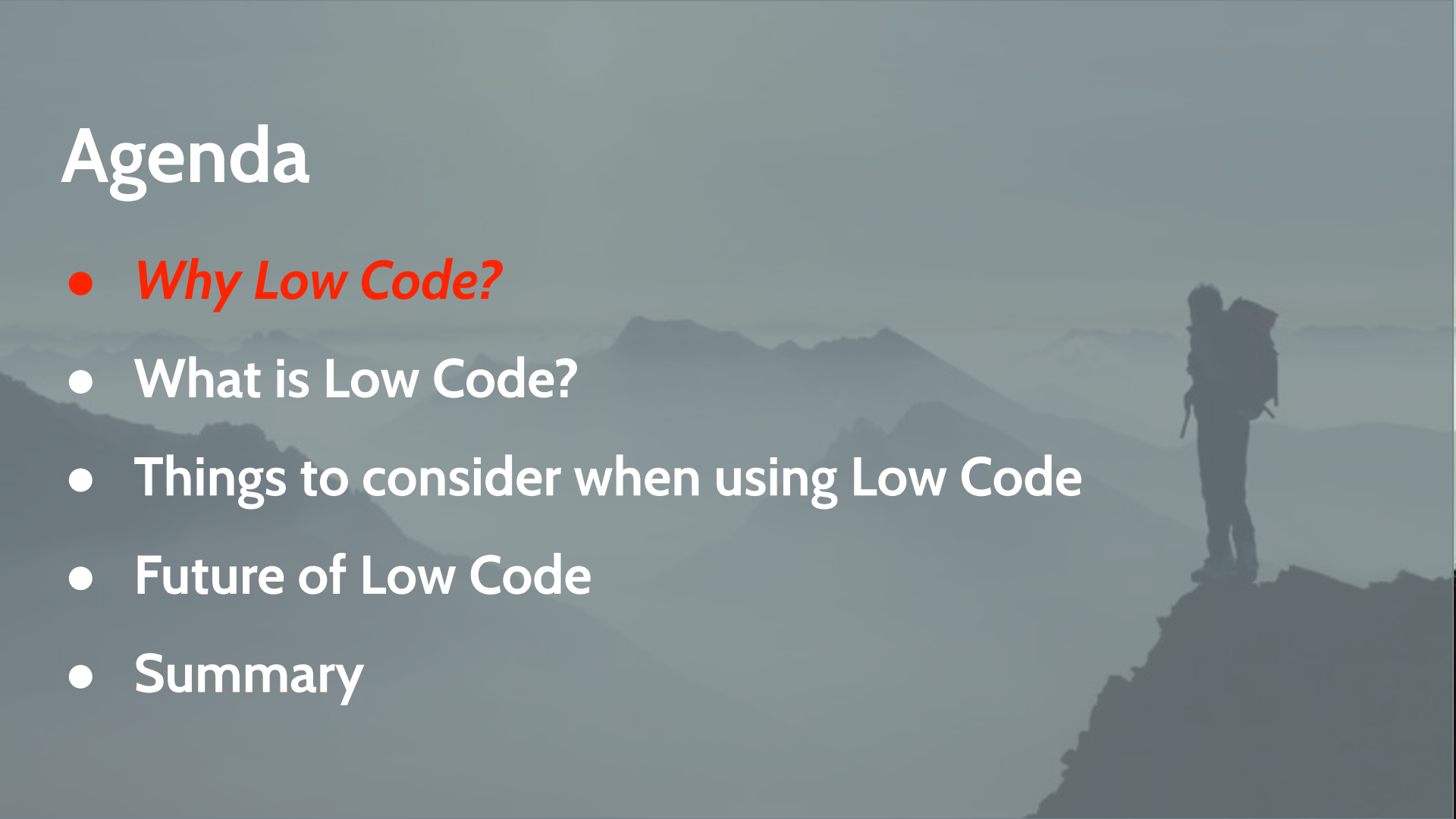
Agenda

- Why Low Code?
- What is Low Code?
- Things to consider when using Low Code
- Future of Low Code
- Summary



Agenda

- *Why Low Code?*
- What is Low Code?
- Things to consider when using Low Code
- Future of Low Code
- Summary



Writing software is still a very **manual process.**

But:

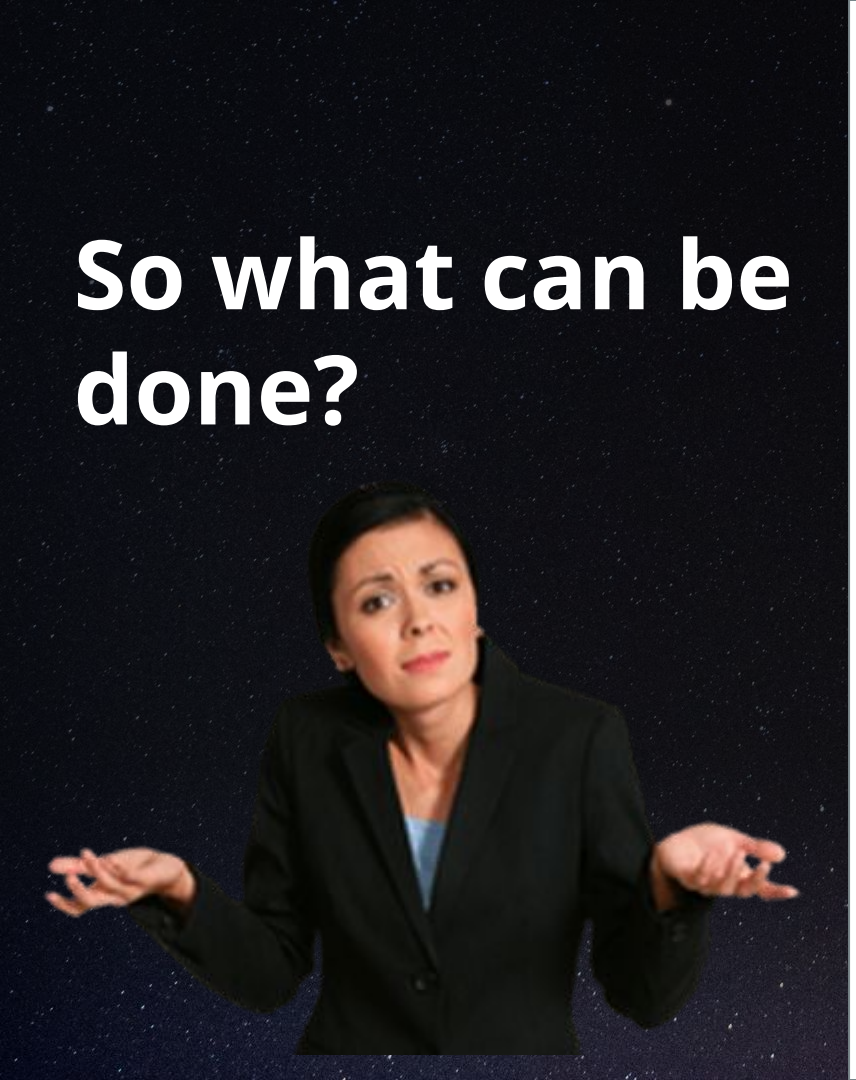
- **500 Mio Apps have to be build in the next 3 years**
  IDC study 2019[1]
  213 Mio companies worldwide

- **By 30 Mio developers today?**
  50 Mio StackOverflow visitors each month
  40 Mio Github users
  24 Mio developers estimated by Evans Data Corporation [2]

- **5 - 6 Apps per Dev per Year?**

(1) https://www.businesswire.com/news/home/20191029005144/en/IDC-FutureScape-Outlines-the-Impact-Digital-Supremacy-Will-Have-on-Enterprise-Transformation-and-the-IT-Industry
(2) https://evansdata.com/press/viewRelease.php?pressID=278

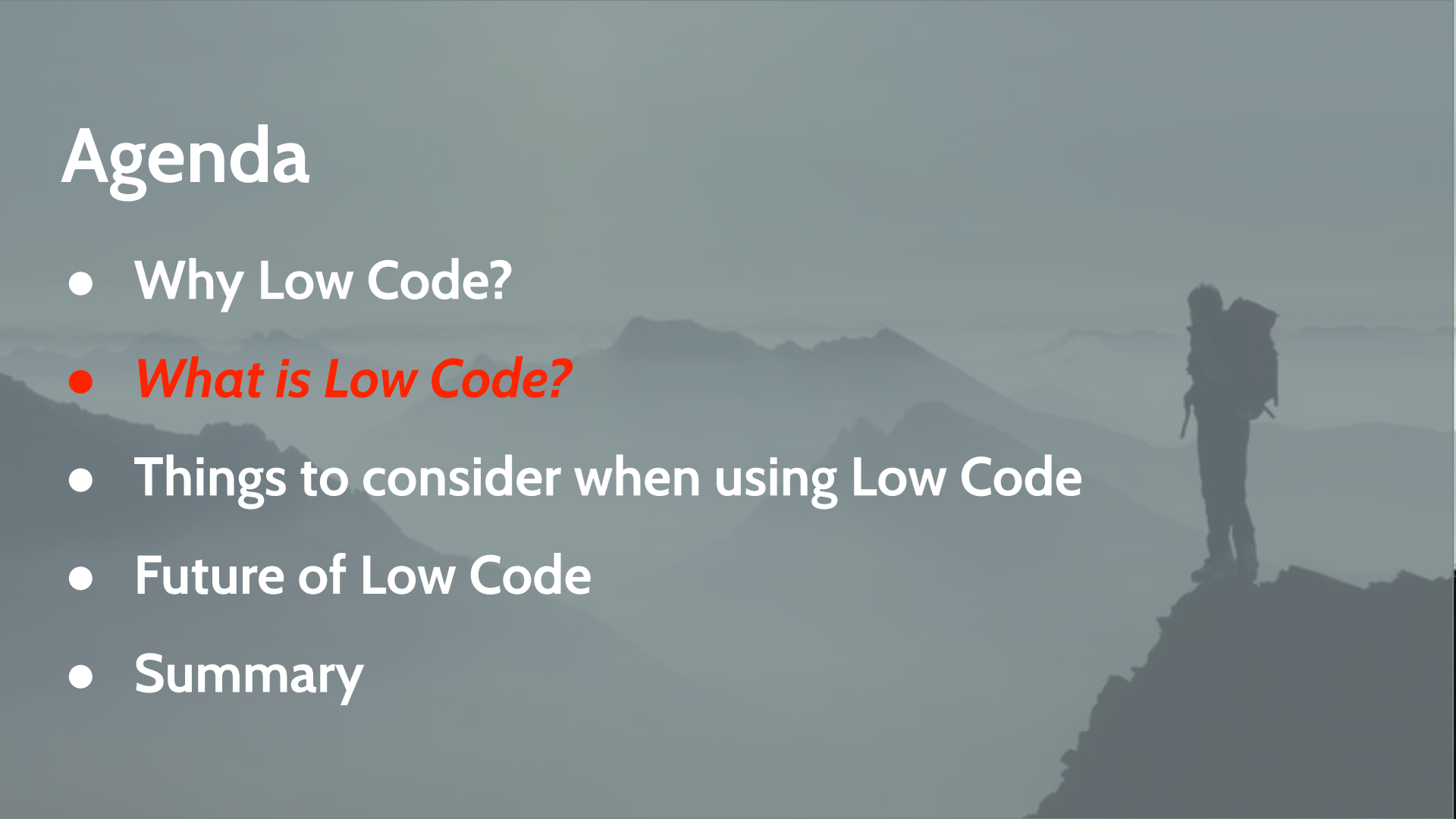
So what can be done?



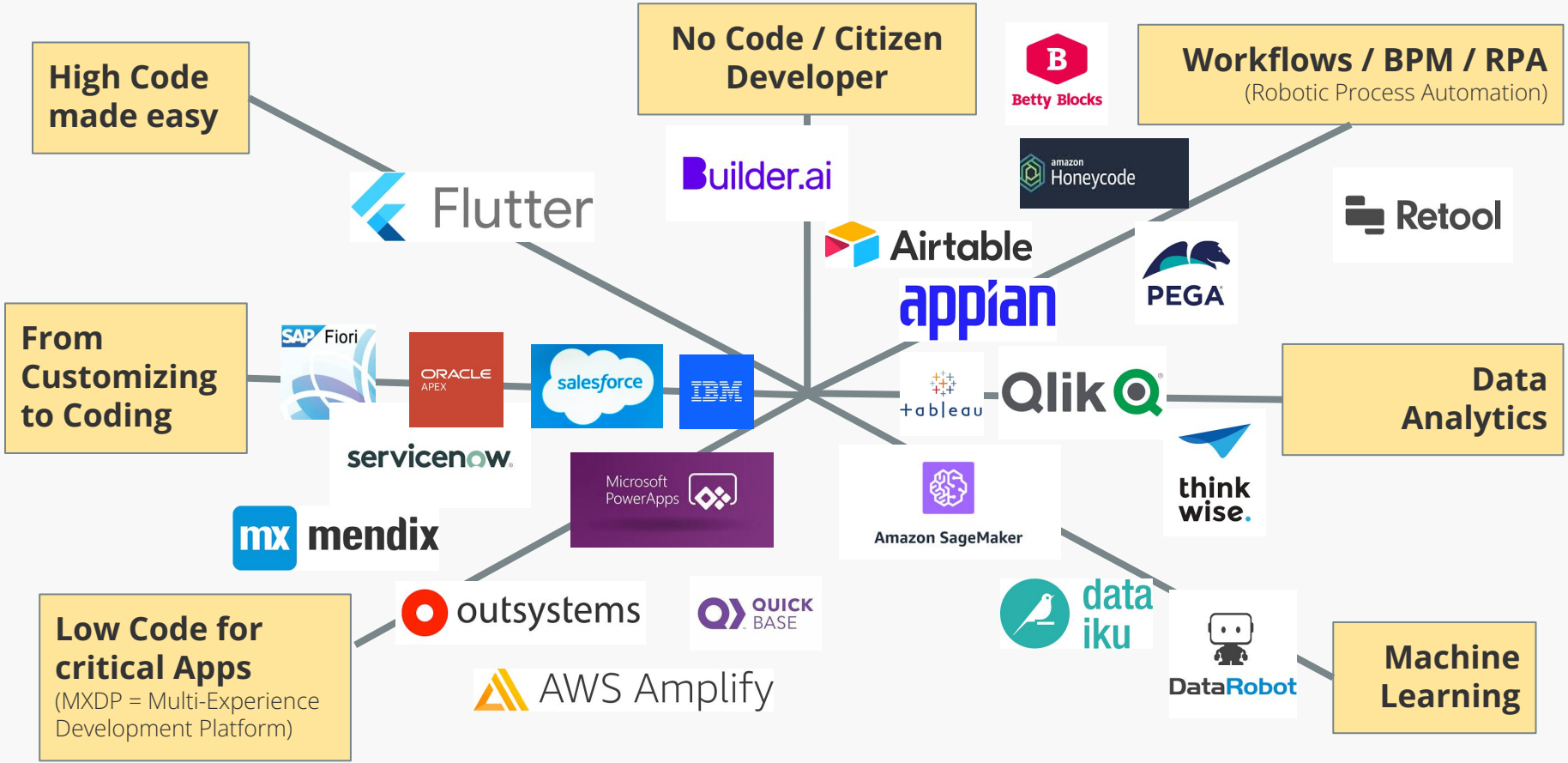
- Make developers more efficient
- Enable more people to develop

Agenda

- Why Low Code?
- *What is Low Code?*
- Things to consider when using Low Code
- Future of Low Code
- Summary



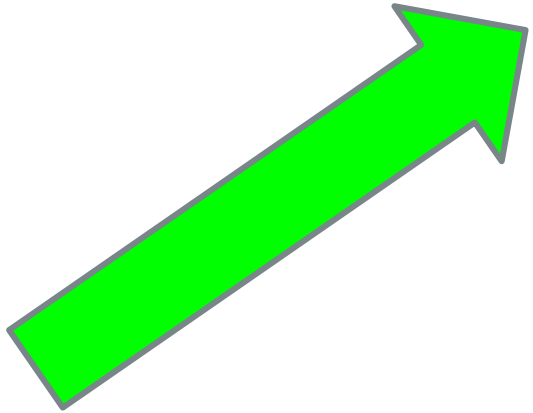
Low Code is emerging everywhere!



The Low Code / No Code market is booming

- **47%** of enterprises are using Low Code / No Code (2021 TechRepublic Survey https://www.zdnet.com/article/survey-low-code-and-no-code-platform-usage-increases/)

- By 2025, **70%** of new applications developed by enterprises will use Low Code or No Code technologies (Gartner LCAP Magic Quadrant 2021)



Low Code is not new

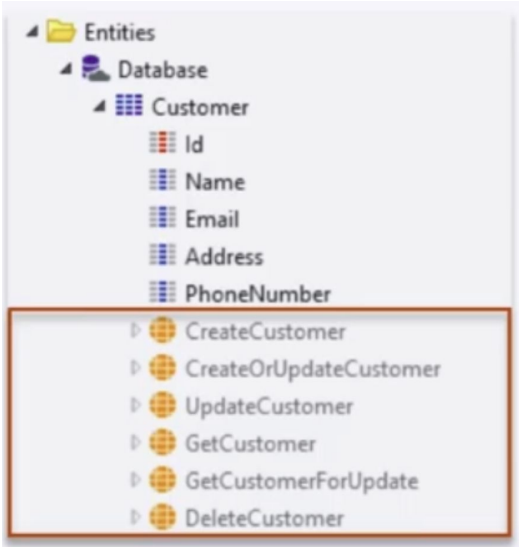
- MS Excel
- Lotus Notes
- Ruby on Rails
- RAD (Rapid Application Development)
- CASE (Computer Aided Software Engineering)
- 4GL (4th generation Programming Languages)
- … and many other examples

What makes Low Code more efficient?

- *Standardization*
- *Abstraction*
- *Visual Programming*

Standardize and pre-define almost everything

- Database, OS, Runtime, Authorization & Security, Servers, Virtualization, Storage, Network as PaaS/SaaS in the cloud
- Software Development Life Cycle (SDLC): pipelines, CI/CD, testing, versioning, orchestration
- Integration with other systems via APIs (REST, SOAP)
- Database CRUD routines
- UI Templates



The picture is a screen shot from OutSystems Service Studio

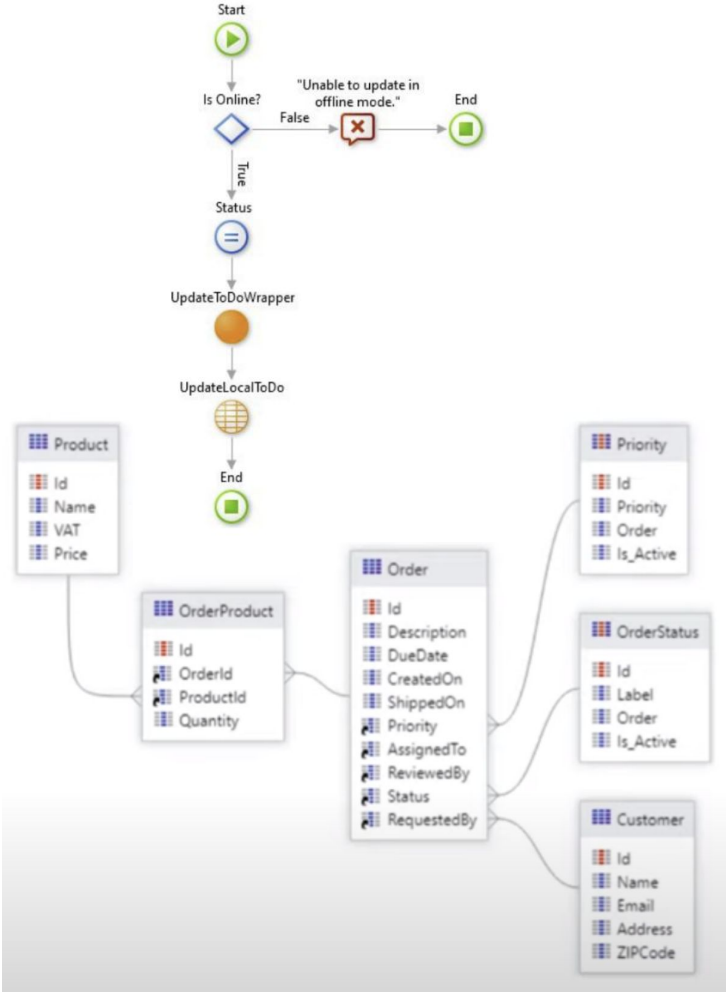
Specify what you want to do, not how to do it

- Long history of increasing abstraction in programming (from assembler to modern domain-specific languages to frameworks like Spark)

The picture is a screen shot from OutSystems Service Studio

Visual Programming

- Visual Programming is not new:
  - 1949: Von Neumann and Goldstine apply flowcharts to describe computer programs
  - 1994: UML (Unified Modeling Language) was developed
- Other professions are using visual specification (architecture, construction, etc.)



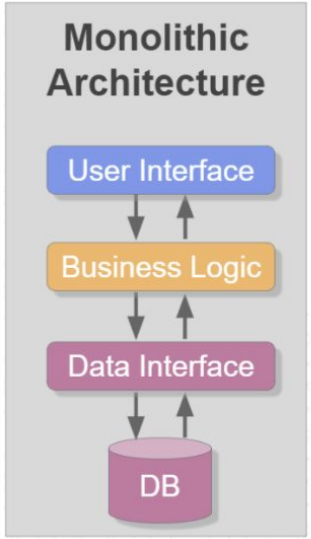
The pictures are screen shots from OutSystems Service Studio

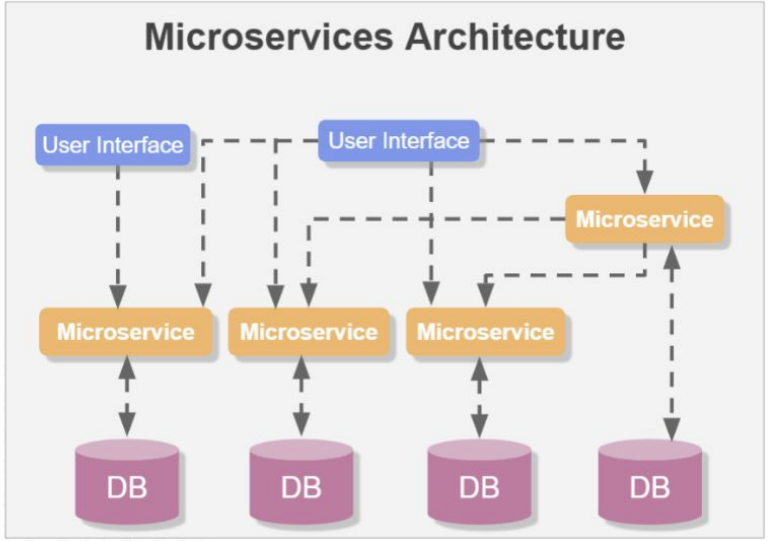
Low Code and modern SW Development

- *Micro-Service Architecture*
- *Test-Driven Development (TDD)*
- *Continuous Delivery*

Micro-Service Architecture

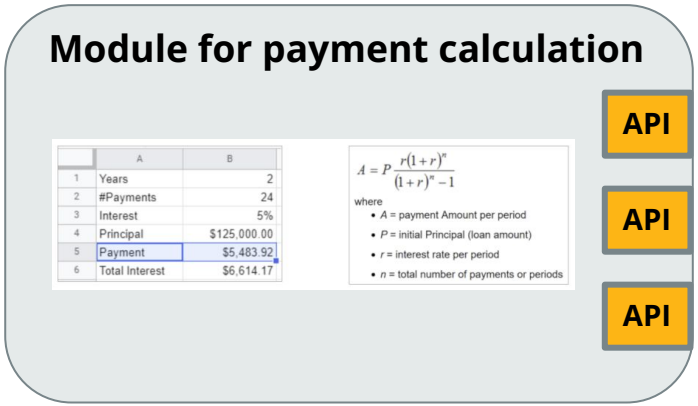
- Functionality is encapsulated in **modules**
- Expose functionality via:
  - **Server Actions** in the same environment
  - **Service Actions** via REST API or SOAP Web Services
- Use Cases for Micro-Service Architecture:
  - Complex or multiple Applications
  - Independent teams
  - Modernize core legacy systems





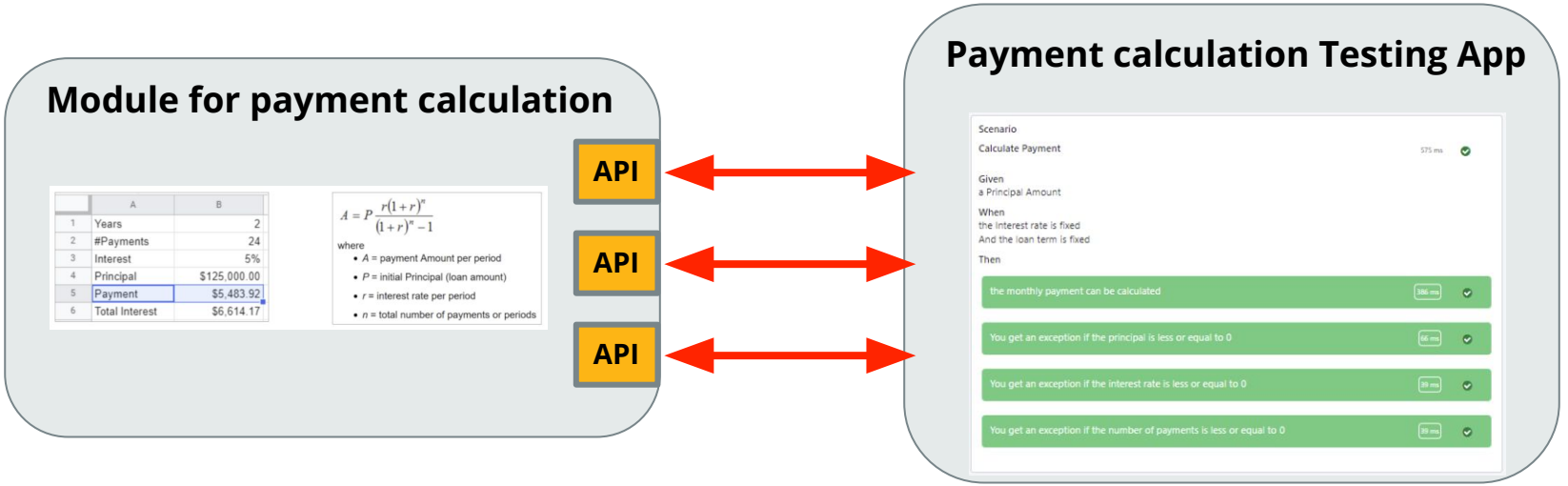
Pictures from:
https://itnext.io/outsystems-microservices-architecture-use-case-edf522608077

Test-Driven Development (TDD)



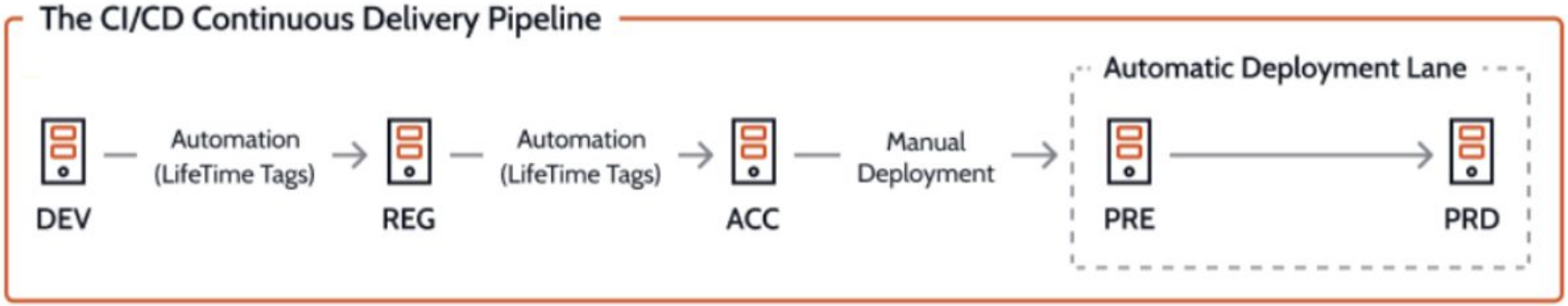
Example from OutSystems Blog "TDD Example: Creating a Loan App": https://www.outsystems.com/blog/posts/test-driven-development-example/

Test-Driven Development (TDD)



Example from OutSystems Blog "TDD Example: Creating a Loan App": https://www.outsystems.com/blog/posts/test-driven-development-example/

Pipelines and Continuous Delivery

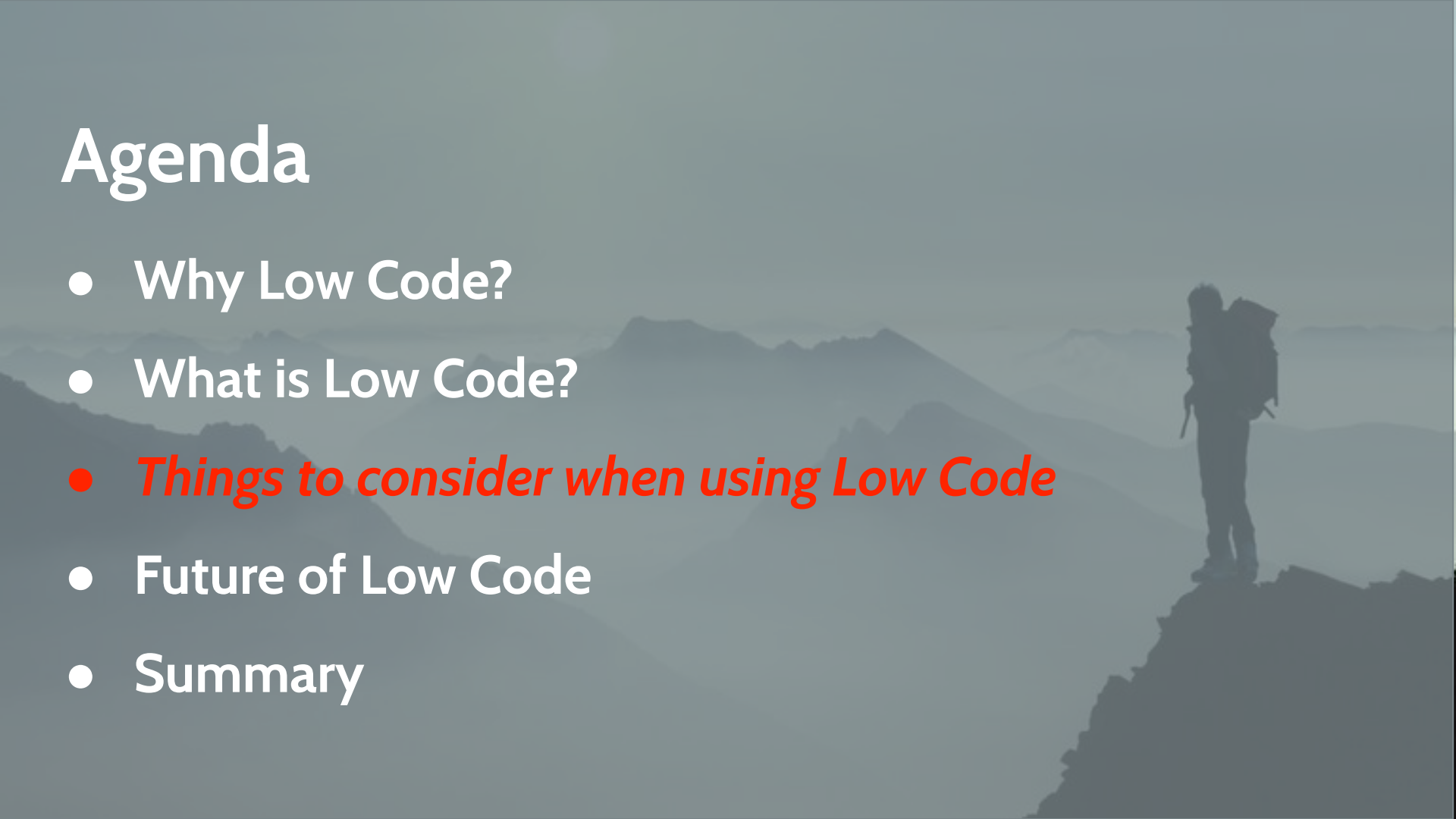


- Several environments connected by pipelines
- Automatic testing, releasing and deployment procedures
- Central system (here: LifeTime) is controlling the processes
- CI/CD orchestrators (ex. Jenkins) can control the pipelines by APIs

Picture from: https://success.outsystems.com/Documentation/How-to_Guides/DevOps/How_to_build_an_OutSystems_continuous_delivery_pipeline

Agenda

- Why Low Code?
- What is Low Code?
- *Things to consider when using Low Code*
- Future of Low Code
- Summary



Things to consider when you are planning to use Low Code

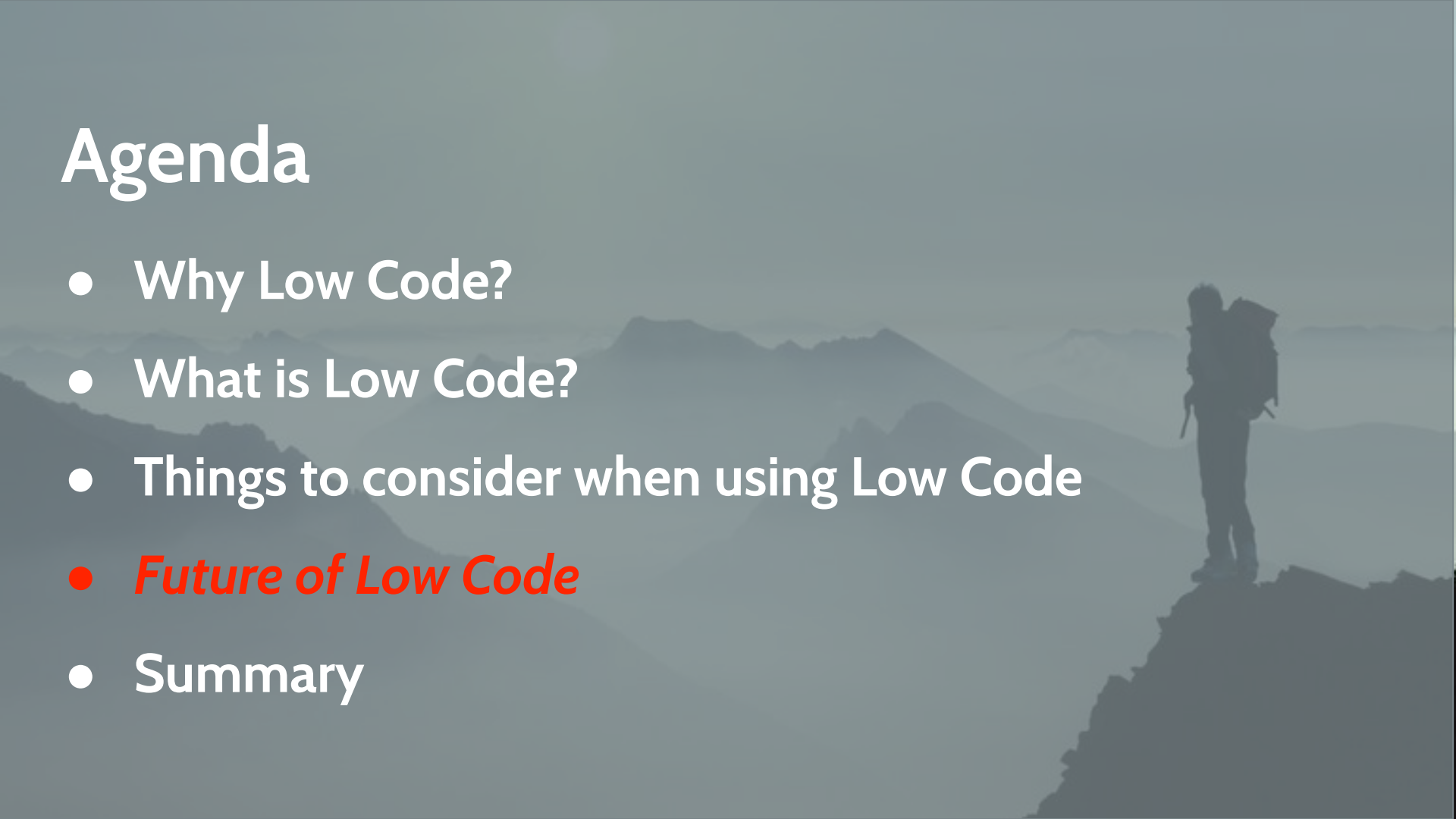
- Low Code developers available?
- Is your Low Code vendor able to maintain APIs over the long run (updates, tests, documentation, etc.)?
- Vendor Lock-In: What is when you want to terminate the contract with the Low Code vendor?
- Are you able to scale if a Low Code app becomes successful and/or mission-critical (in terms of number of users, requirements, integration)?
- Risk of becoming the next "legacy system"?
- SDLC does not scale as well with #devs and app complexity as high code

Things to consider when you are planning to use No Code

- Shadow-IT must be managed!
- Define guidelines for app and release management, user management, authorizations, security, access to APIs , backups, etc.
- On-board and train citizen developers
- Set-up No Code support organization or CoE (Centre of Excellence)

Agenda

- Why Low Code?
- What is Low Code?
- Things to consider when using Low Code
- *Future of Low Code*
- Summary

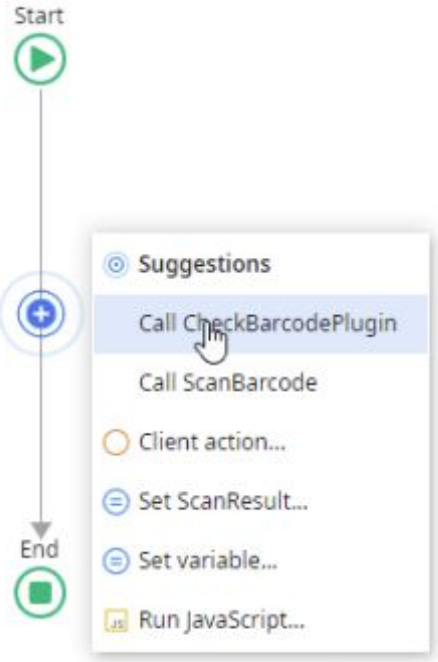


AI-Assisted Development

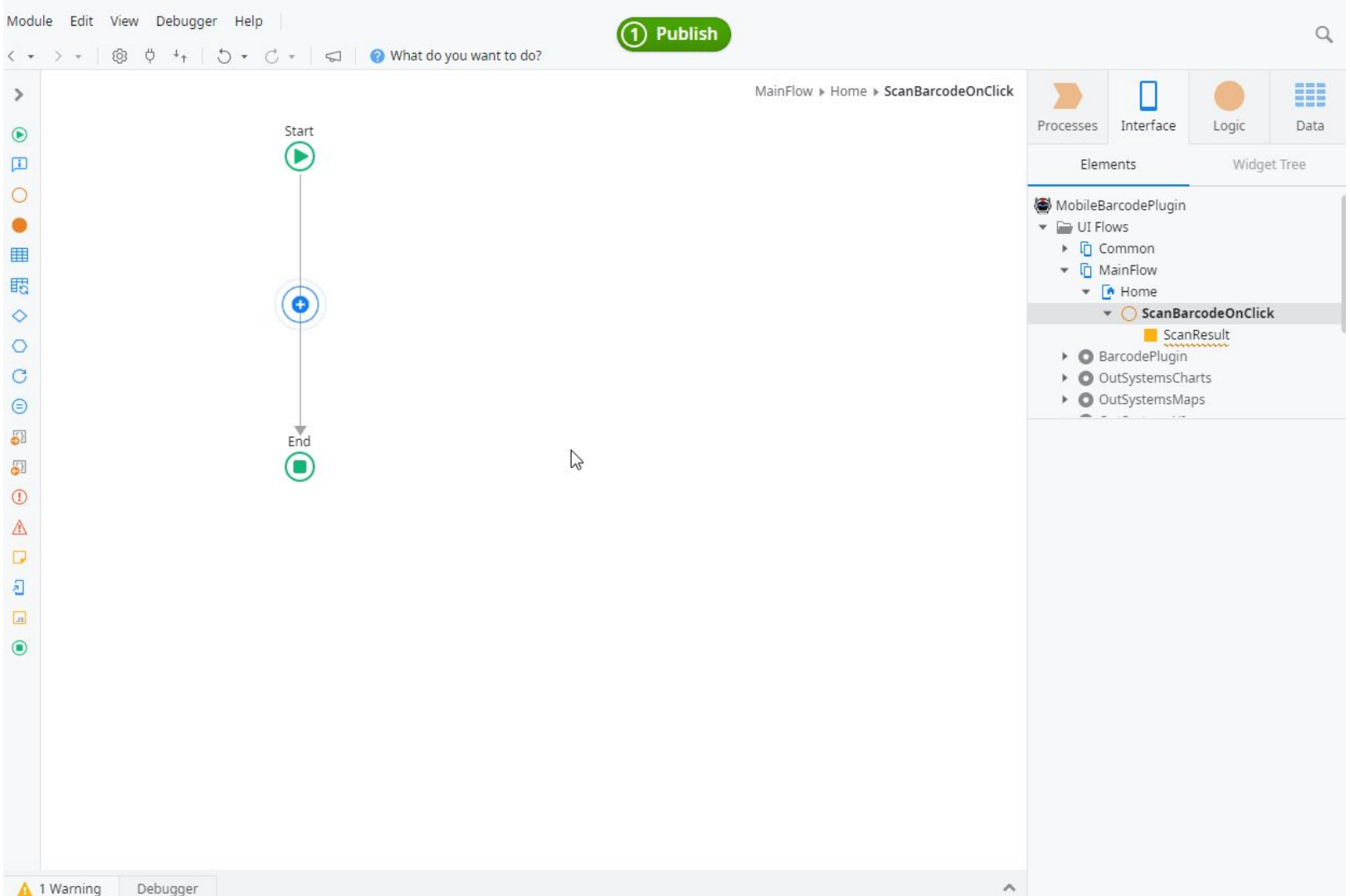
- Assistants trained on millions of anonymized code samples do code suggestions on the fly

- Can significantly save development time

⇒ **See demo on next slide**



Pictures from: https://success.outsystems.com/Documentation/11/Developing_an_Application/Implement_Application_Logic/AI-assisted_development


Module
Edit
View
Debugger
Help
1 Publish
What do you want to do?
MainFlow ▸ Home ▸ ScanBarcodeOnClick
Start
End
Processes
Interface
Logic
Data
Elements
Widget Tree
MobileBarcodePlugin
UI Flows
Common
MainFlow
Home
ScanBarcodeOnClick
ScanResult
BarcodePlugin
OutSystemsCharts
OutSystemsMaps
1 Warning
Debugger

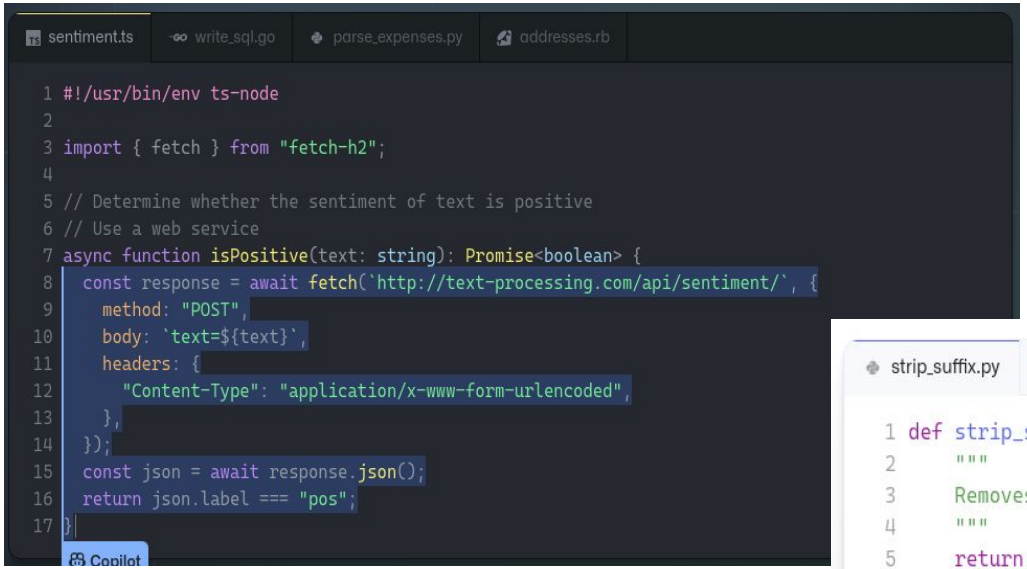
Large Language Models (LLMs) are used for Coding
Example 1: GitHub Copilot

- Developed with OpenAI Codex / GPT3
- Trained on public repositories on GitHub and other publicly available sources
- Works in Python, JavaScript, TypeScript, Ruby, Java, Go, React and many other languages
- Publicly available as Technical Preview
- Intelligent Auto-Complete

(https://copilot.github.com/)

GitHub Copilot
Service

GitHub Copilot - Example:



Large Language Models (LLMs) are used for Coding Example 2: DeepMind's AlphaCode

- Trained on GitHub and Exercises from Codeforces (https://codeforces.com/)
- Creates Python and C++ programs
- Achieves a rank within the top 54% of human participants in Codeforces competitions

(https://www.deepmind.com/blog/competitive-programming-with-alphacode)

AlphaCode

DeepMind's AlphaCode - Example:

You are given two strings s and t, both consisting of lowercase English letters. You are going to type the string s character by character, from the first character to the last one.

When typing a character, instead of pressing the button corresponding to it, you can press the "Backspace" button. It deletes the last character you have typed among those that aren't deleted yet (or does nothing if there are no characters in the current string). For example, if s is "abcbd" and you press Backspace instead of typing the first and the fourth characters, you will get the string "bd" (the first press of Backspace deletes no character, and the second press deletes the character 'c'). Another example, if s is "abcaa" and you press Backspace instead of the last two letters, then the resulting text is "a".

Your task is to determine whether you can obtain the string t, if you type the string s and press "Backspace" instead of typing several (maybe zero) characters of s.

Input

The first line contains a single integer q ($1 \le q \le 10^5$) — the number of test cases.

The first line of each test case contains the string s ($1 \le |s| \le 10^5$). Each character of s is a lowercase English letter.

The second line of each test case contains the string t ($1 \le |t| \le 10^5$). Each character of t is a lowercase English letter.

It is guaranteed that the total number of characters in the strings over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print "YES" if you can obtain the string t by typing the string s and replacing some characters with presses of "Backspace" button, or "NO" if you cannot.

You may print each letter in any case (YES, yes, Yes will all be recognized as positive answer, NO, no and nO will all be recognized as negative answer).

Input

```
4
ababa
ba
ababa
bb
aaa
aaaa
aababa
ababa
```

AlphaCode's Output:

```
t=int(input())
for i in range(t):
    s=input()
    t=input()
    a=[]
    b=[]
    for j in s:
        a.append(j)
    for j in t:
        b.append(j)
    a.reverse()
    b.reverse()
    c=[]
    while len(b)!=0 and len(a)!=0:
        if a[0]==b[0]:
            c.append(b.pop(0))
            a.pop(0)
        elif a[0]!=b[0] and len(a)!=1:
            a.pop(0)
            a.pop(0)
        elif a[0]!=b[0] and len(a)==1:
            a.pop(0)
    if len(b)==0:
        print("YES")
    else:
        print("NO")
```

Output

```
YES
NO
NO
YES
```

Pictures from: https://www.deepmind.com/blog/competitive-programming-with-alphacode

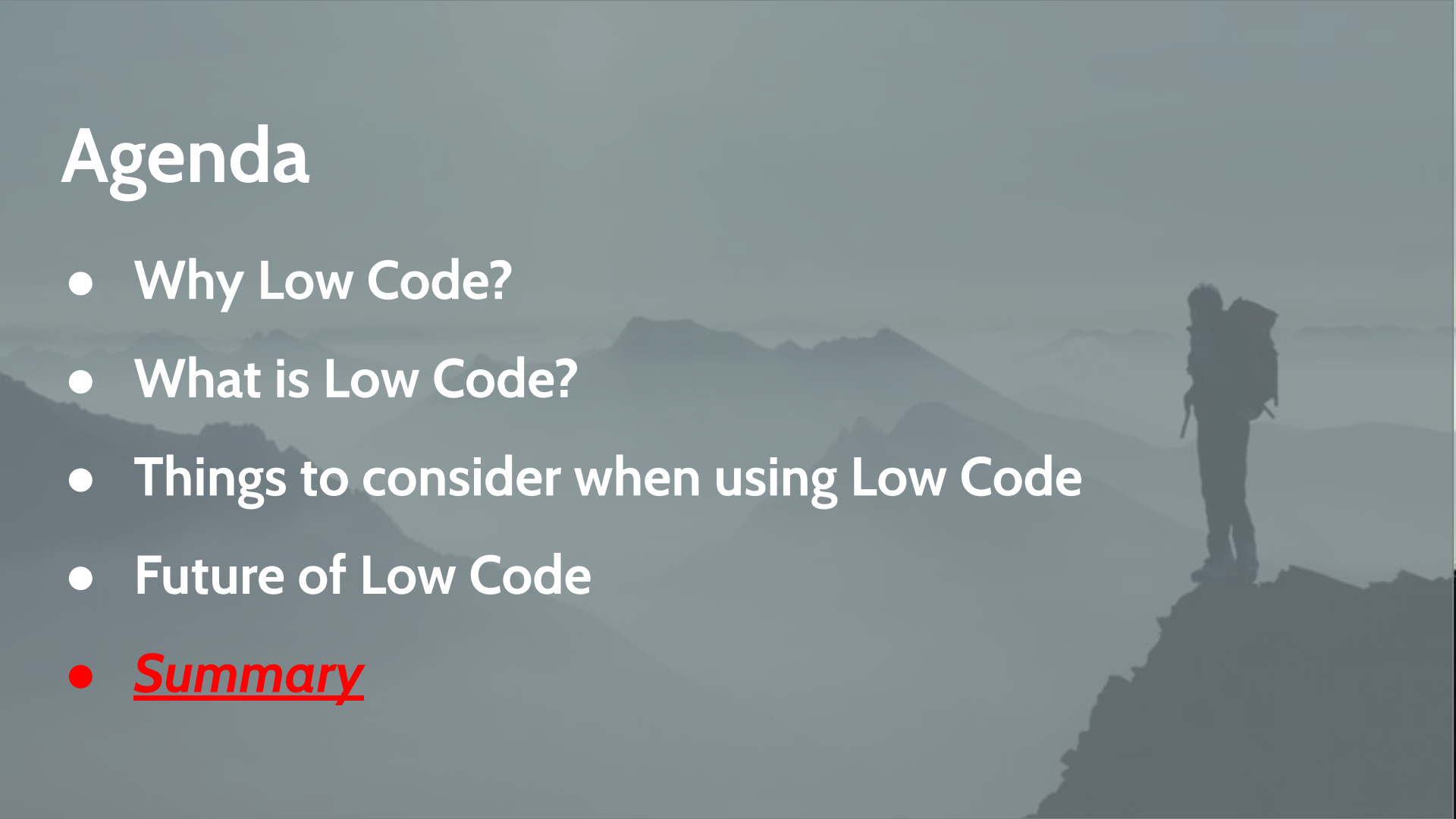
Risks and Limitations of Using Language Models for Coding

- Language Models for coding are far from perfect!
  GitHub Copilot: 43% correct in the first try, 57% correct in 10 attempts(*)
- The suggested code is not tested, it might not even compile or run or makes sense
- Suggestion based on a limited context (few hundred lines max.)
- May suggest old or deprecated functions or libraries
- Code might be insecure
- Suggested code can contain personal data (email addresses, phone numbers, access keys, etc.)!

(*) Source: https://copilot.github.com/

Agenda

- Why Low Code?
- What is Low Code?
- Things to consider when using Low Code
- Future of Low Code
- ***Summary***



Summary: Advantages of using Low Code

- It is not a thread - it is a tool box!
- Boost your efficiency as a developer!
- Great for rapid prototyping
- Closer cooperation with the business
- Develop once for different platforms (web, mobile)
- *Democratizing Development!*

Thank You!

Christoph Windheuser, OutSystems
GOTO Amsterdam
June 15, 2022

GOTO
Guide

★★★★★

Remember to **rate this session**

THANK YOU!

Download on the App Store

GET IT ON Google Play

#GOTOams