System

Module A

Mo

per -10

!

per -6

per -4

?

per -2

?

+10 at completion

+10 at

dule B

Module C

!

!

?

?

?

completion

+10 at completion

Place card

Dependencies

System

Module A

Mo

per -10

!

per -6

per -4

?

per -2

?

+10 at completion

+10 at

dule B

Module C

!

!

?

?

?

completion

+10 at completion

Place card

Dependencies

User Count

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99
100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139
140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179

Final Scoring

	Number of users	completed modules	TD in Architecture (Row 1)	TD in Feature (Row 2)	TD in Feature (Row 3)	TD in Feature (Row 4)	Plus Points	Minus Points	Total
Team 1		x10=	x10=	x6=	x4=	x2=			
Team 2		x10=	x10=	x6=	x4=	x2=			

Overview

a) Process ticket

* Roll two dice:

- Take on TD (opt.) and →
- Doubles? →
- For each matching die →

- **Ticket complete?:**
  - Place the current ticket on the board
  - !/?-field?: Draw a face-down !/?-card
  - Record points (Only if)
  - Draw a new ticket. Place a workstation.
  - Read aloud the !/?-card and play the ?-card immediately. ! ?
    !-card can be played at an advantageous moment at the end of a turn.

Current Ticket

OR

b) Reduce TD

- Flip the workstation to 'Reduce TD'
- Determine TD
- Roll a die
  In : 4, 5 or 6
  In : 5 or 6
- success?: Reduce TD → and flip the workstation

Overview

a) Process ticket

* Roll two dice:

- Take on TD (opt.) and →
- Doubles? →
- For each matching die →

- **Ticket complete?:**
  - Place the current ticket on the board
  - !/?-field?: Draw a face-down !/?-card
  - Record points (Only if)
  - Draw a new ticket. Place a workstation.
  - Read aloud the !/?-card and play the ?-card immediately. ! ?
    !-card can be played at an advantageous moment at the end of a turn.

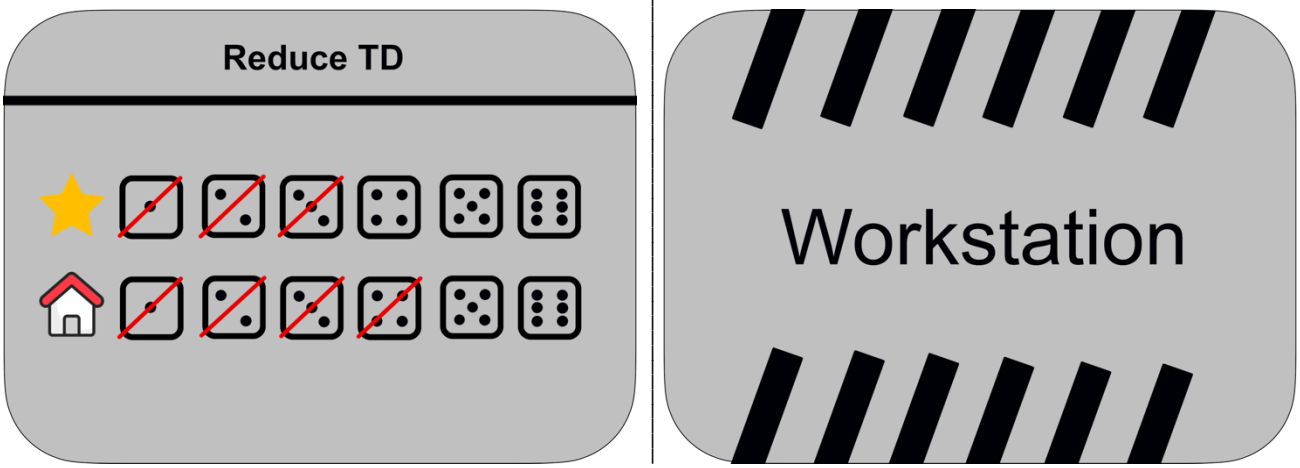
Current Ticket

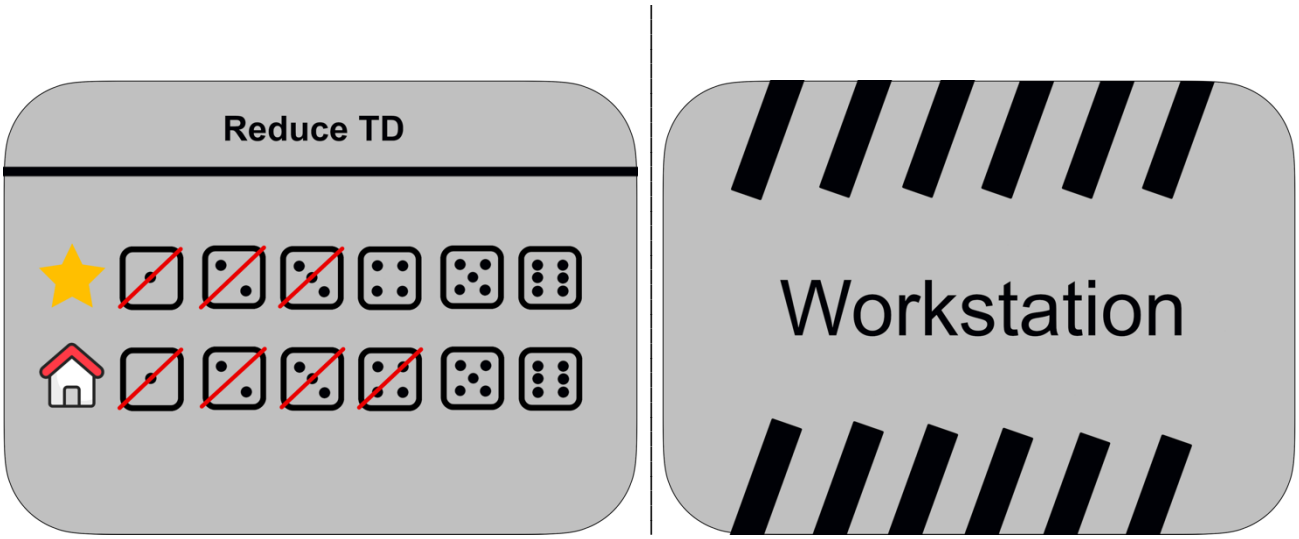
OR

b) Reduce TD

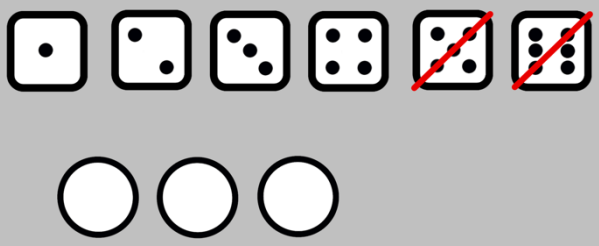
- Flip the workstation to 'Reduce TD'
- Determine TD
- Roll a die
  In : 4, 5 or 6
  In : 5 or 6
- success?: Reduce TD → and flip the workstation

Fold line

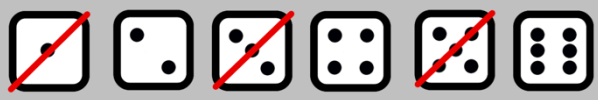




Broker



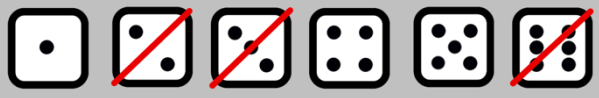
Blockchain



Client Server

Domain Driven Design

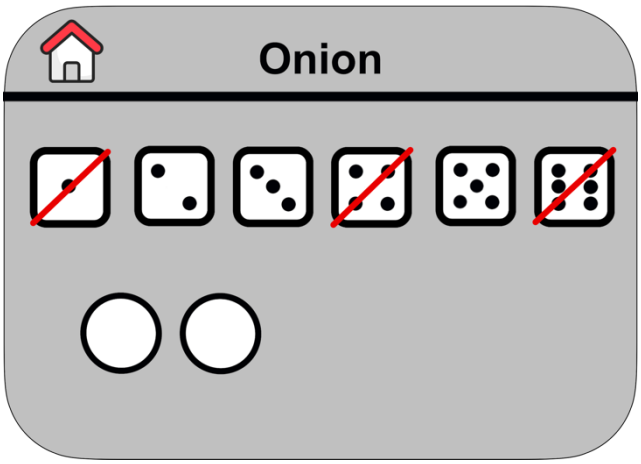
Edge Computing

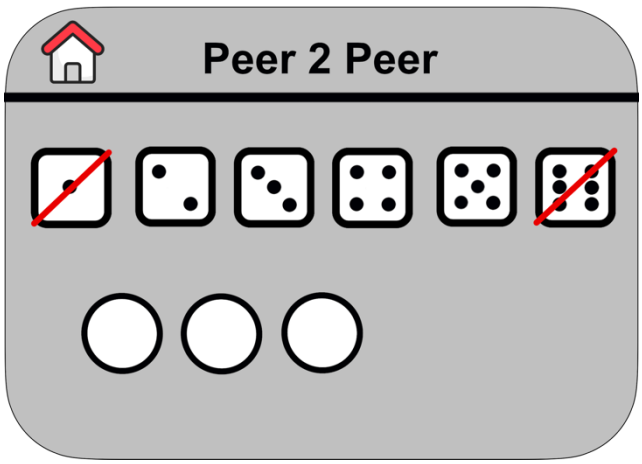


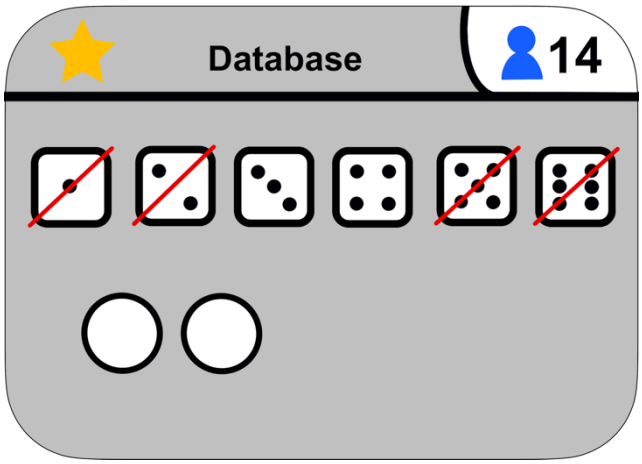
Layer

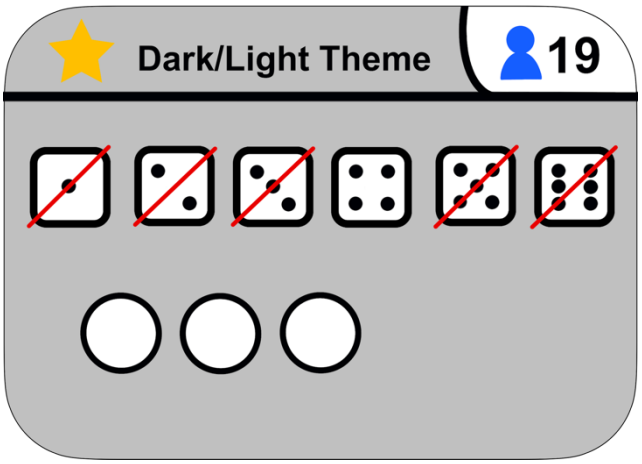
Microkernel

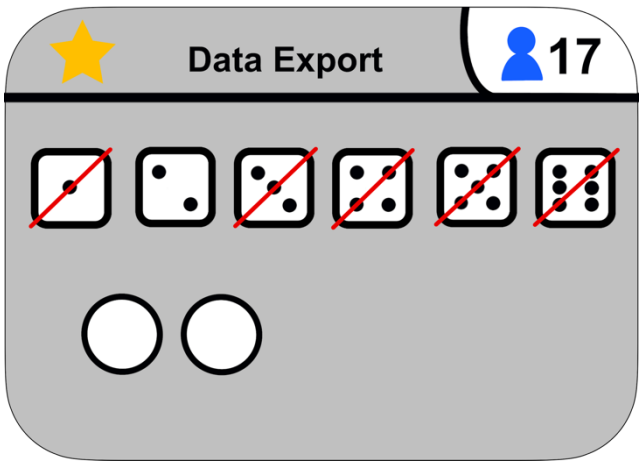
Model View Controller

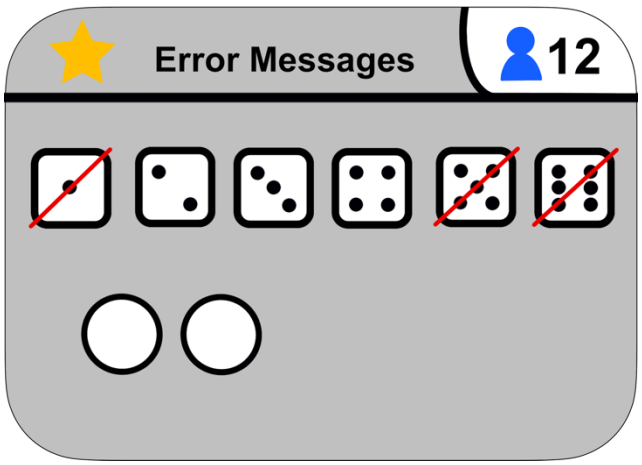

Onion

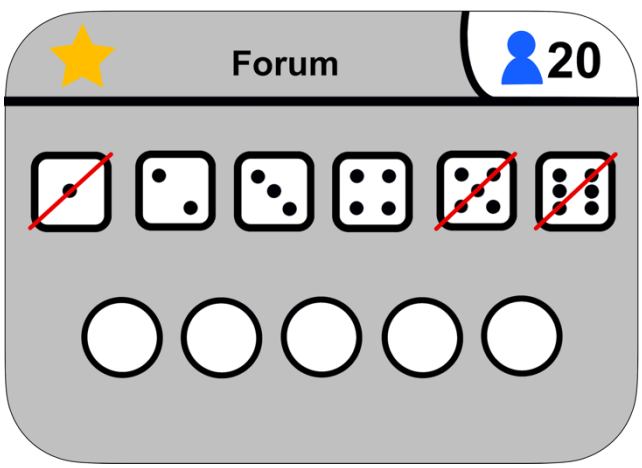

Peer 2 Peer

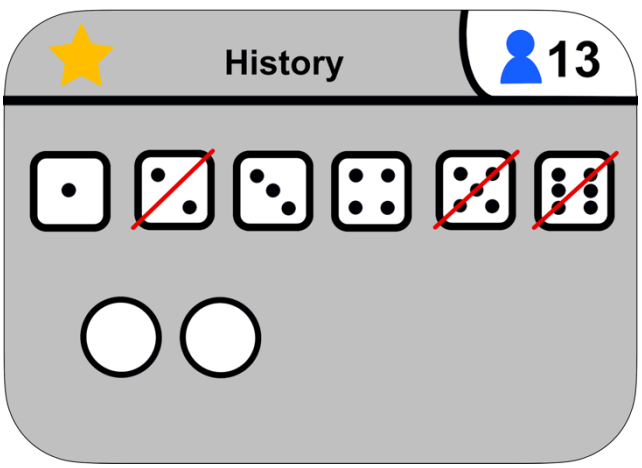

Database
14

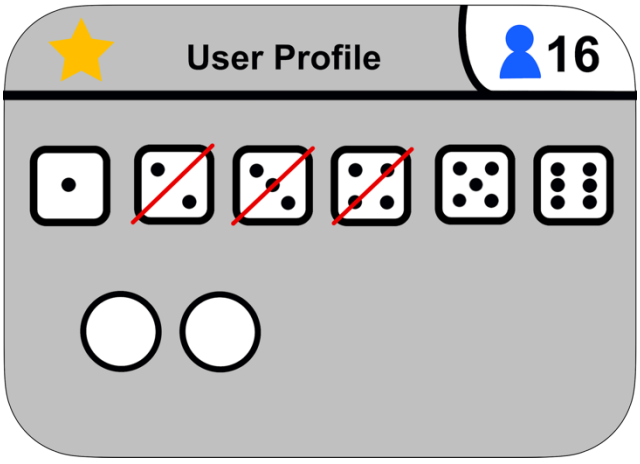

Dark/Light Theme
19

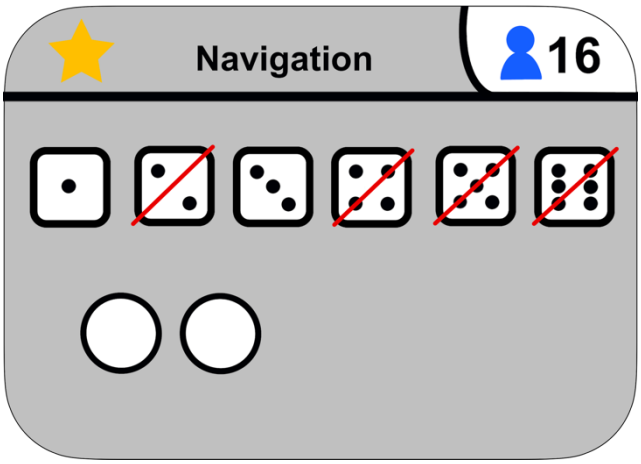

Data Export
17

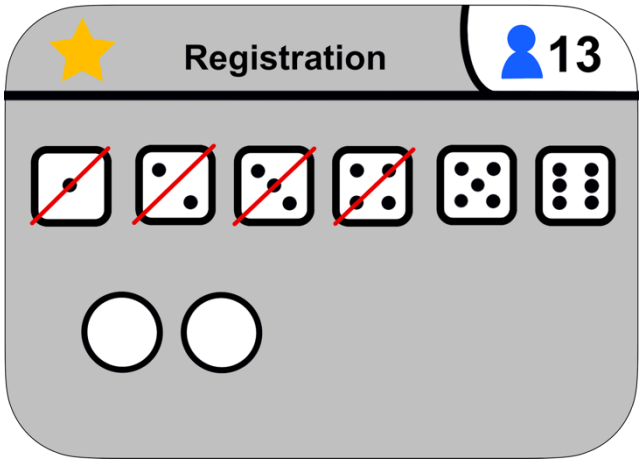

Error Messages
12

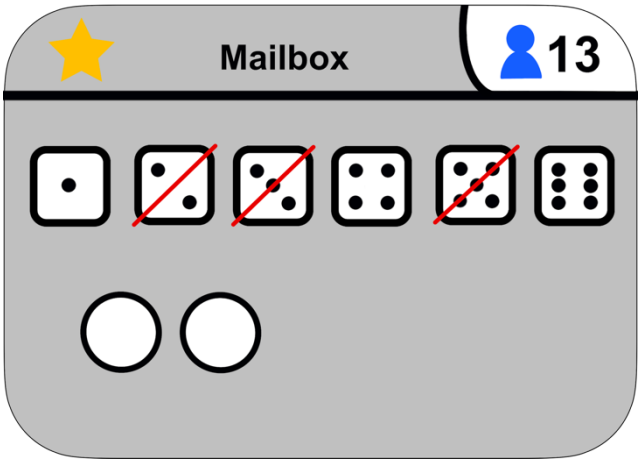

Forum
20

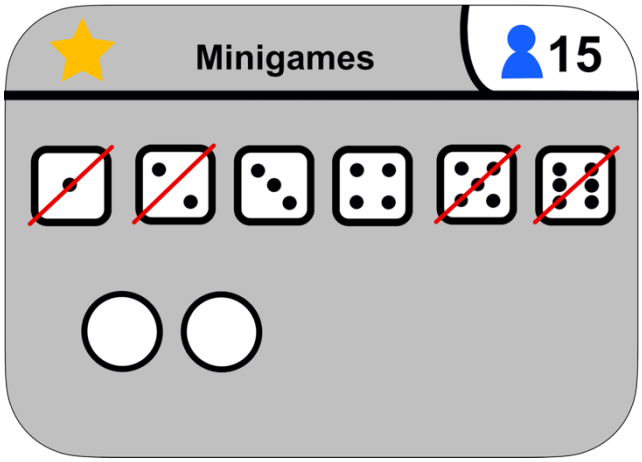

History
13


User Profile
16


Navigation
16


Registration
13


Mailbox
13


Minigames
15

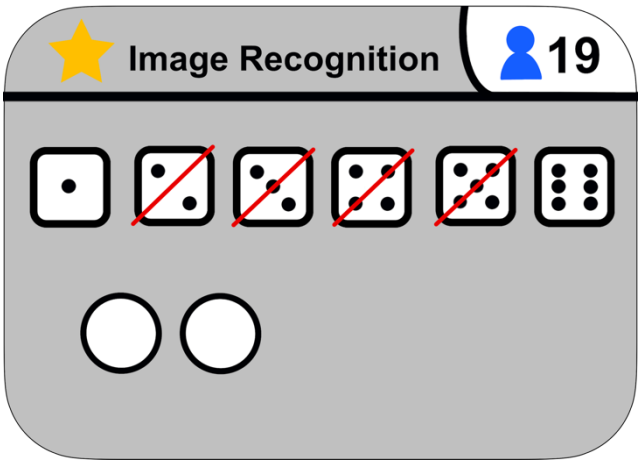
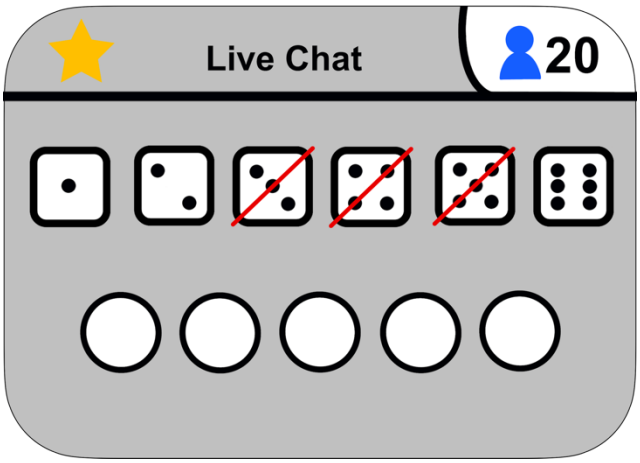
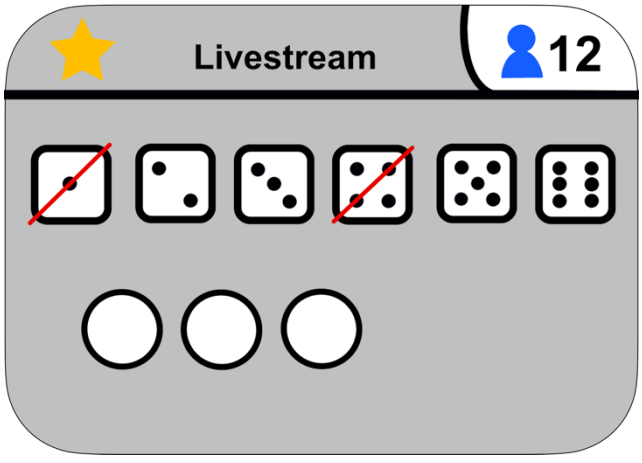
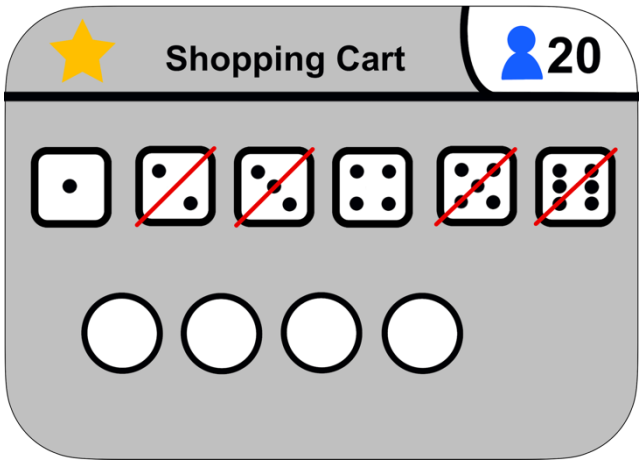
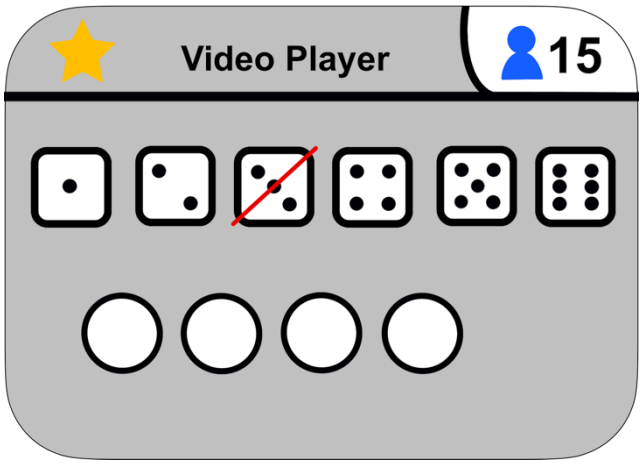


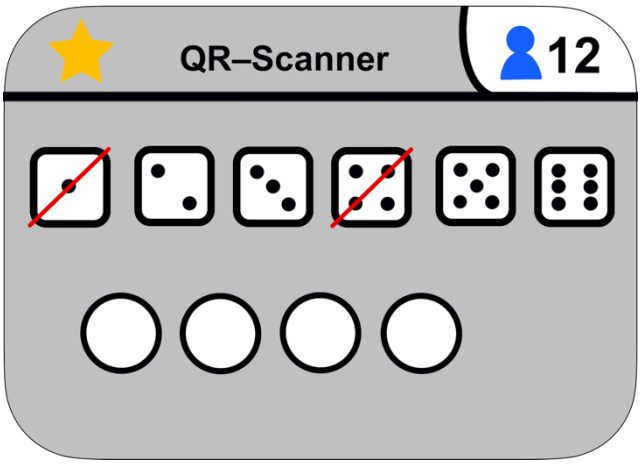
Image Recognition
19

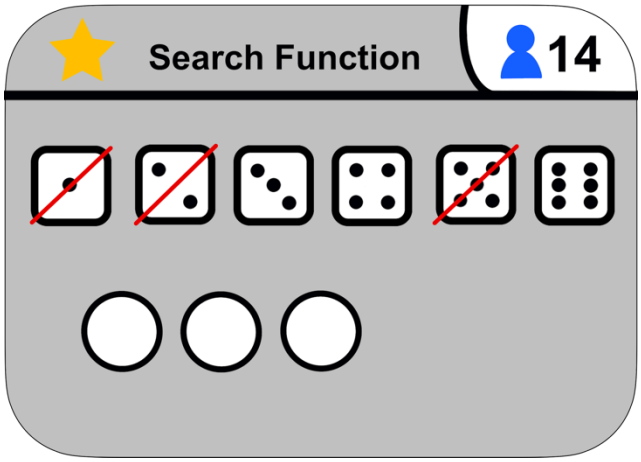

Live Chat
20

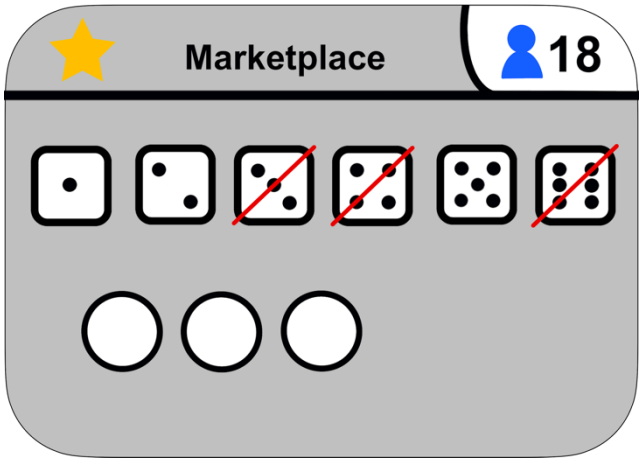

Livestream
12

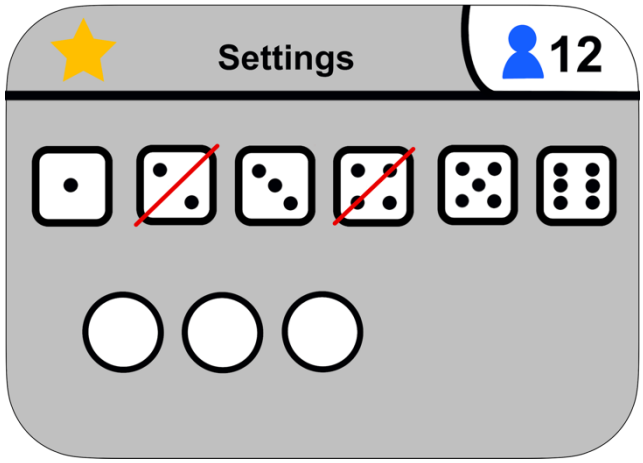












! Engage Specialists

Bringing in specialists can uncover and resolve previously unknown problems.

You can remove any TD.

! Effective Involvement

By getting involved in the programmers challenges, management becomes aware of current issues in ongoing development. Management trusts the developers and is ready to invest more time in quality.

You can take an extra turn.

! Experience

By communicating with programmers from other projects, the origin of TD can be determined and resolved based on experience.

You can remove any TD.

! Clarify the Wh-Questions

Properly identifying technical debt through the Wh-Questions helps to infer the actual technical debts from symptoms. It becomes easier to eliminate technical debts.

You can remove any TD.

! Play This Game

Educating team members about technical debt can prevent poor prioritization and project failure.

You can take an extra turn.

! More Personnel

Management recognizes that many developers are overloaded. To better distribute the workload, management decides to hire new developers.

You can take an extra turn.

! Hire a Working Student

Management decides to hire a working student. They can take on simple tasks, allowing developers more time for complex tasks.

You can take an extra turn.

! Refactoring

A developer is assigned to appropriately divide overly long classes and methods. This reduces the complexity of the project.

You can remove any TD.

! Team Communication

Team communication can make members aware of the impacts of technical debt and that it should be effectively managed.

You can remove any TD.

! Uncover the Origin

Repeatedly asking about the cause helps uncover the origins of technical debt.

You can remove any TD.

? Architectural Decisions C

You realize how much the quality of your system depends on architectural decisions.

If you have three or more TD in your system, your opponent can place a TD on any architecture ticket.

? Increased Costs C

Technical debts have led to performance losses. To optimize performance, the servers are being parallelized, leading to additional licensing costs.

If you have two or more TD in your system, your opponent can place a TD on any feature ticket.

? Internal Vicious Circle C

Developers previously chose a provisional solution that led to unreadable code. This has now created a vicious circle where developers opt for more provisional solutions, causing more unreadable code.

If you have two or more TD in your system, your opponent can place a TD on any feature ticket.

? Loss of Control C

By overplanning for too many possibilities, too many unnecessary options were implemented. The project has become hard to manage. Management is struggling to regain control.

If you have two or more TD in your system, your opponent can place a TD on any feature ticket.

? Missing Test Framework C

Setting up the test framework takes too much time, so it is initially skipped. Every new line of code now creates technical debt because the tests will have to be created later, once a framework is in place.

Your opponent can place a TD on any feature ticket.

? Missing Functions C

The customer complains about missing functions in the system. Due to too many TD, these functions cannot be implemented.

If you have three or more TD in your system, your opponent can place a TD on any architecture ticket.

? Neglected Test Maintenance C

To save time and resources, test requirements and the test database are not regularly updated. This makes the system error-prone and increases the number of post-release bugs, leading to customer frustration and potentially damaging the company's reputation.
If you have two or more TD in the system, skip a turn.

? Unmanageable Test Debts C

The accumulated test debts are no longer manageable. Unstable or outdated tests undermine confidence in the test results and cause error messages to be ignored. As a result, important errors are overlooked, leading to system crashes. The team has to spend intensive time on bug fixing and has no time to implement new features.
You skip a turn.

? Stress and Resignations C

Management has incurred TD that are not planned to be repaid. As a result, developers have more long-term work and are stressed, causing some to even leave the company.

If you have two or more TD in your system, skip a turn.

? Chain of Consequences C

Due to a missed deadline, developers now have to work overtime. The stress this causes leads them to look for someone to blame. However, this unproductive behavior only results in even more time pressure.

Any ticket in your system with two or more TD gets an additional TD that your opponent can place.

? Chain of Causes C

Due to time pressure, there is a lack of documentation. This lack of documentation in turn makes it difficult for new developers to familiarize themselves with the system, leading them to incur further technical debt.

Any ticket in your system with two or more TD receives an additional TD that your opponent can place.

? Vacation C

There isn't enough time to document the code for the current ticket. The only developer familiar with the ticket is on vacation. You must wait for the developer's return.

If you have two or more TD in your system, skip a turn.

? Communication Problems C

The customer demands a reprioritization of the current ticket. However, management communicates this to the developers too late. They must incur TD to implement it on time.

Swap your current ticket for a new one from the deck and immediately place a TD on it.

? External Vicious Circle C

There are too few developers in the company, leading them to handle too many tasks. This causes stress among some developers, prompting them to leave the company.

Any ticket in your system with two or more TD receives an additional TD that your opponent can place.

? Developer Retires C

A skilled developer retires. Although they wrote good code, it was poorly documented. Successors must invest a lot of time to understand their code.

You skip a turn.

? Outdated Technology C

The technology chosen by management at the start has become outdated. Despite this, management does not want to update it. Implementing new features takes increasingly longer.

Your opponent can place a TD on any feature ticket.

? Legislative Change C

A legal change that takes effect next month necessitates a modification to the system. Unfortunately, management learned about this law too late, and the change must now be handled with the highest priority.

Swap your current ticket for a new one from the deck and immediately place a TD on it.

? Invisible Debts C

The system appears to function flawlessly from the outside, even though many technical debts have been accumulated. Since there are currently no customer complaints, management decides to continue taking shortcuts in implementation.

Your opponent can place a TD on your current ticket.

? I Don't Want This Anymore! C

The customer saw how the latest ticket will look during a demo and doesn't want it anymore. Instead, another ticket must be implemented as quickly as possible, necessarily incurring TD.

Swap your current ticket for a new one from the deck and immediately place a TD on it.

? Lack of Awareness C

Management misunderstands TD as a temporary trend and misses the opportunity to address the issue properly.

Your opponent can place a TD on your current ticket.

? Financial Pressure C

The current ticket requires additional licensing costs, exceeding the project's budget. To adhere to the budget, a cheap and provisional solution must be found.

Your opponent can place a TD on your current ticket.

? Poor Requirements C

Poorly documented requirements from the customer led to the wrong architectural choices.

Your opponent can place a TD on any architecture ticket.

? Time Pressure C

To meet the deadline for the current ticket, a provisional solution must be implemented.

Your opponent can place a TD on your current ticket.