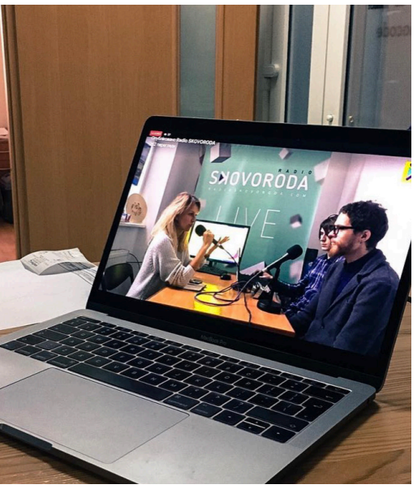
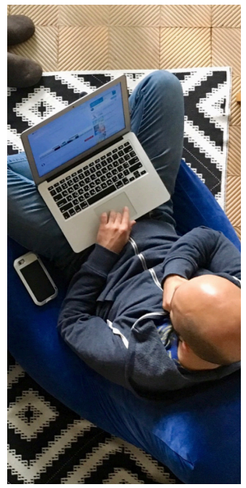
innocode
create relevance

Digital Products Development

We have helped startups and enterprises launch and grow digital products since 2010

INNOCODE.COM





Content

Design and development methodologies 4

Culture and recruitment 7

TicketCo Case Study 8

Workflow 10

VALERII'S CORNER

WE CARE ABOUT YOUR PRODUCT

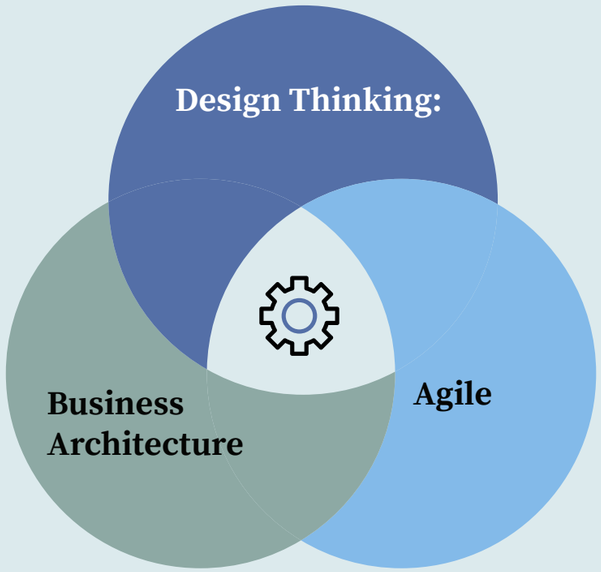
When we started Innocode in 2010 we wanted to use common sense as much as possible. With the growth of Agile development methodologies, people were already saying that 'technology' is only part of the success when you build a new digital product. No matter how good the technology is, or how good your delivery skills are, it doesn't matter if you are building the wrong thing. How do you define what is the right thing to build? First, you ask 'Why' very often. Engineers are often obsessed with the question 'How', for example: how to build this and that, how that will work together, etc. Managers are often thinking 'What', 'what is the scope, what is the budget, what is the bonus'. But in order to make the right decisions, we need to have an environment with a good level of transparency, trust and knowledge sharing.

An environment where everyone has a shared context on why we are building something, and also the tools, processes and methodologies that allows us to move the discussion from subjective to objective measures. This has become a part of our culture, something deeply rooted in our processes, daily operations and rituals.

Innocode is a unique company in the sense that we are running our own product department and selling our products globally, but also building digital products for our selected partners and clients. Such an approach is beneficial both for us and for our partners. Because we have a deep understanding of both positions, we continuously learn how to be a better trusted partner.

Valerii Shypunov
CTO og Partner

Design and Development Methodologies



Design Thinking:

puts the needs of the end user at the forefront
Advantages:
- End user and subject matter expert defined requirements
- Top of the knowledge funnel
- Everything is possible

Disadvantages:
- Does not focus on cost or schedule
- Can set expectations incorrectly

Agile:

puts rapid delivery of functionality and updates first
Advantages:
- True rapid deployment of functionality and fixes
- Responsiveness

Disadvantages:
- Can result in MVP being"too minimal"

Business Architecture:

puts business needs and constraints first
Advantages:
- Better understanding of real boundaries
- Forces financial analysis

Disadvantages:
- Internally focused
- Limits thinking

Each of the methodologies has their advantages and disadvantages, so we are not treating a particular methodology as the 'silver bullet', instead we are building cross-functional teams sufficient enough in the methodologies specified to be able to use the advantages of one in order to cover the disadvantages of another.

While **design thinking** and **agile development** methodologies competence were acquired during the process of building digital products for Innocode's clients and partners, **business architecture** competence was built when Innocode started building their own digital products in 2015.

How are the design and development methodologies implemented in the process?

The most common first step in the collaboration on a new concept is a pre-project. The goal of the pre-project is to build context awareness and reduce uncertainty.

Workshop

We believe it is extremely important to meet people that you are going to work with face to face. All our pre-projects starts with a workshop, either at the client's premises or an Innocode office.

Why is this important?

1. To find out whether we are compatible or not. Even though it's about all the digital product development, there are human beings on both sides of this process, and sometimes people are compatible, sometimes they are not. Better to find this out as soon as possible.
2. A workshop with all participants in the same room provides the possibility to focus on the business context and transfer the knowledge more efficiently. Communication is also more effective.

The goal of the workshop is to brainstorm and document as much knowledge about the business domain, context, market and identified hypothesis as possible.

Workshop duration: 1-3 days.

Pre-project research

After the workshop, the team will work iteratively on conducting the business and technical research. It will usually cover the following areas:

- Business model research
- Competition research
- Technological viability research
- Risk analysis

The main goal of the research is to identify the core hypothesis of the new digital product and the most efficient way of validating this hypothesis.

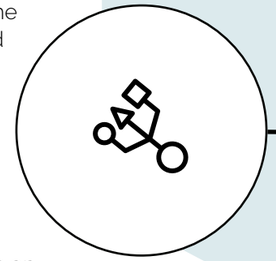
When the core hypothesis is identified, the team will prepare the scope for the minimum viable product (MVP), both in the form of high-level requirements and a set of low-fidelity and high-fidelity prototypes. It is very important to identify the metrics and scenarios on how the core hypothesis will be validated, so it will be possible to pivot at the right time, if needed.
After the pre-project research, Innocode will provide the pre-project report.

Pre-project report artifacts

- Risk analysis report
- High-level requirements
- Ballpark estimates (we use a flexible model of risk handling in the estimates in preparation for scope changes)
- Success scenario and KPI definitions
- Team composition
- Cooperation model (billing, communication plan, responsibilities of Innocode and the client)
- Design
  - Low-fidelity prototype to evaluate interaction flows
  - High-fidelity prototype screens in Figma
  - Animations and transitions effects, close to the real behavior of the product/service

Pre-project values for clients:

- Cost savings via fast validation of the product's core hypothesis
- Client has no commitments for further development phases and can decide to postpone the delivery, proceed with other vendors, etc.



REDUCE UNCERTAINTY

Product Development

After the project kick-off, a cross-functional team (defined during the pre-project) will handle the product development.

Agile development practices presume that the scope might change along the way, thus the separate process of grooming and preparing the scope going in parallel with the actual development and delivery process, because these 2 processes have different pace. In the development process we are following standard Agile development methodologies, like Scrum and Kanban. However, our focus is not limited to just writing the code, but also on the user experience design, practices on delivering, launching and scaling the product for the market.

We are also handling the control function on the:

- Scope
- Timeline
- Budget

Also we emphasize the importance of regular workshops during the development process (once per 6-12 months), in order to:

- Understand long-term plans and priorities for the development
- Perform further risk analysis

Analytics

How do you measure the success of the new digital product? Definitely not by the amount of code delivered or tasks closed in the project tracking software. First of all, you measure it by whether the core hypothesis is validated (i.e. there is a product market fit). In order to identify that on a scale, you need to have the metrics identified, and also captured. We ensure the required data is available at the right time in the right format.

Also, we provide the technical analytics (related to the product stability), based on the logs and customer support requests.

Delivery

Quite often when you work with a custom software development provider, you are focusing on delivering the scope of the first product version, but the actual product delivery to the market is not covered sufficiently.

With competence in running and ensuring stable operation of both the client's products and Innocode own digital products, we have developed the following competences in regards to the delivery process (after development is completed):

- Performance, load and end-user testing
- Preparing documentations (both end-user and technical documentation)
- Preparing training sessions
- Competence on running the live product operations
  - Hosting environment
  - Monitoring
  - Support routines
  - Status page's setup and configuration
- Experience of collaboration with your client success team
  - Analysis of the new product requests
  - Resolving incidents and support cases
- On-site support possibilities

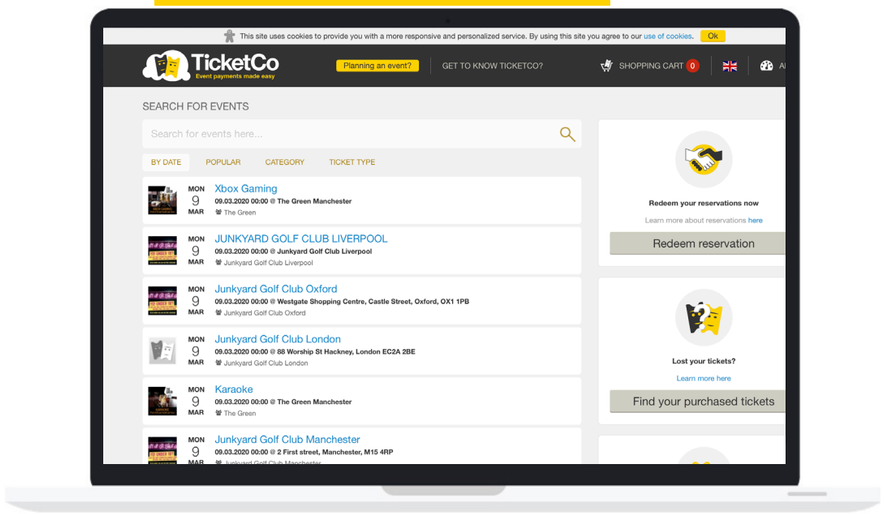
Culture and Recruitment

For Innocode, providing superior quality and value over time is more important than growing the headcount. We are hiring people to grow with Innocode and not just for the particular one-time project. Our recruitment process is multistage, involving multiple stakeholders and even a test-day.

Zero-waste is an important part of the Innocode culture, and it is not limited to physical waste, but waste of time as well. When you are hiring someone, people can prepare and make a great impression in the interview, but when you actually start working with them, it doesn't always work out well. So we decided to create the test-day, where the person can spend a whole day working on the actual tasks with the actual team. During lunchtime, the person also gets to spend time with their other colleagues. Changing jobs is an important decision, and we want to be open and transparent to help people understand if Innocode is a good fit for them, and also gives the team a chance to evaluate their future colleague.

Being open and proactive, involved and curious, trying to resolve issues and not just being protective what makes Innocode employees different.

Case Study:
ticketco



Innocode built an R&D department for TicketCo in 2012, deeply integrated into the TicketCo's organization.

About our cooperation

Our cooperation with TicketCo started in 2012, and as most good things - by coincidence. A friend of a friend introduced us to Kaare Bottolfsen, TicketCo founder and CEO. It was a perfect match because we immediately felt that we had common values: deep involvement of team members into the business context, direct communication and transparency.

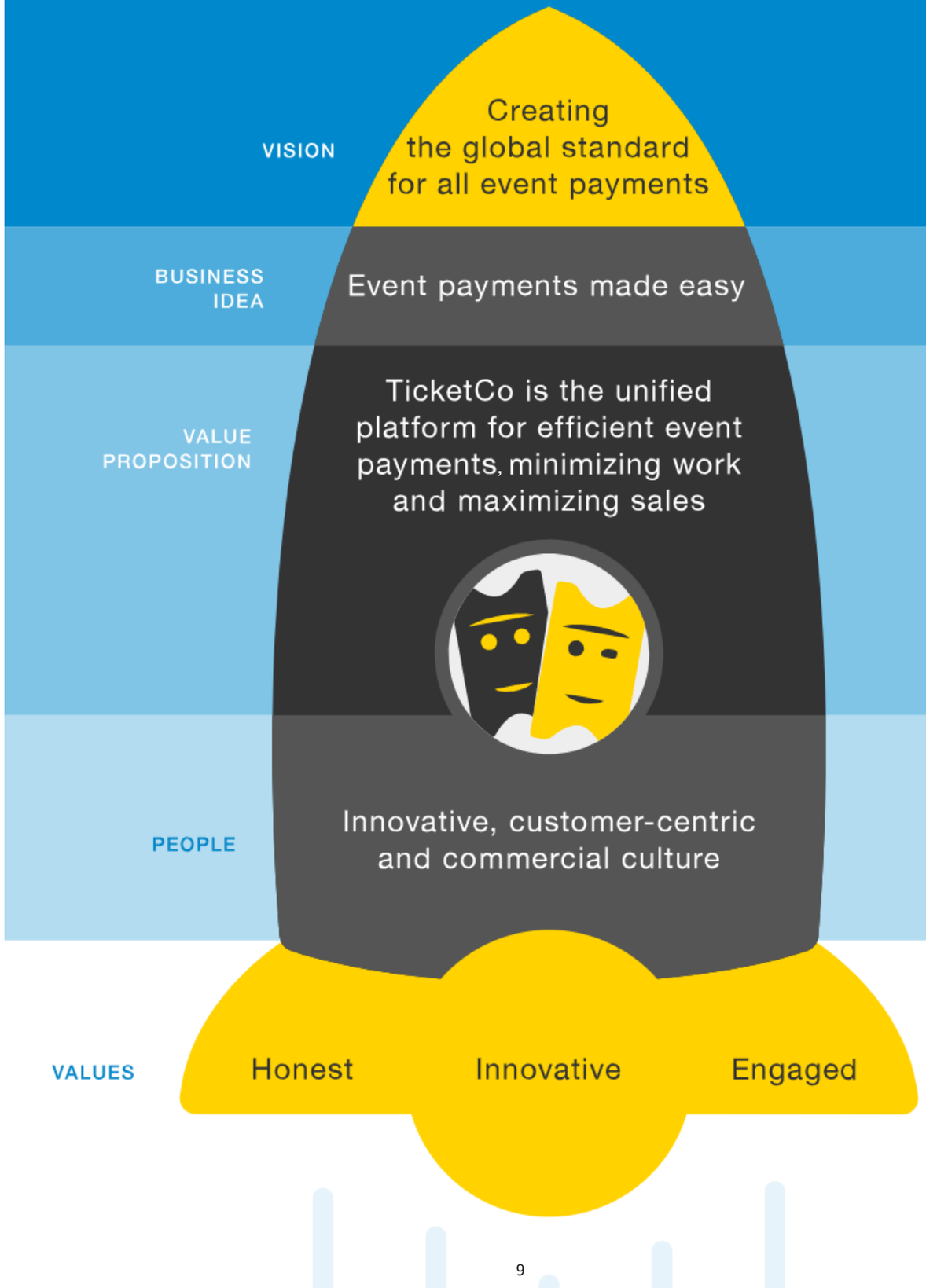
TicketCo has been growing together with Innocode, customer success worked directly with the developers. When TicketCo started serving some of the largest festivals in Norway, Innocode's R&D team visited the festivals to test out the support process in person.

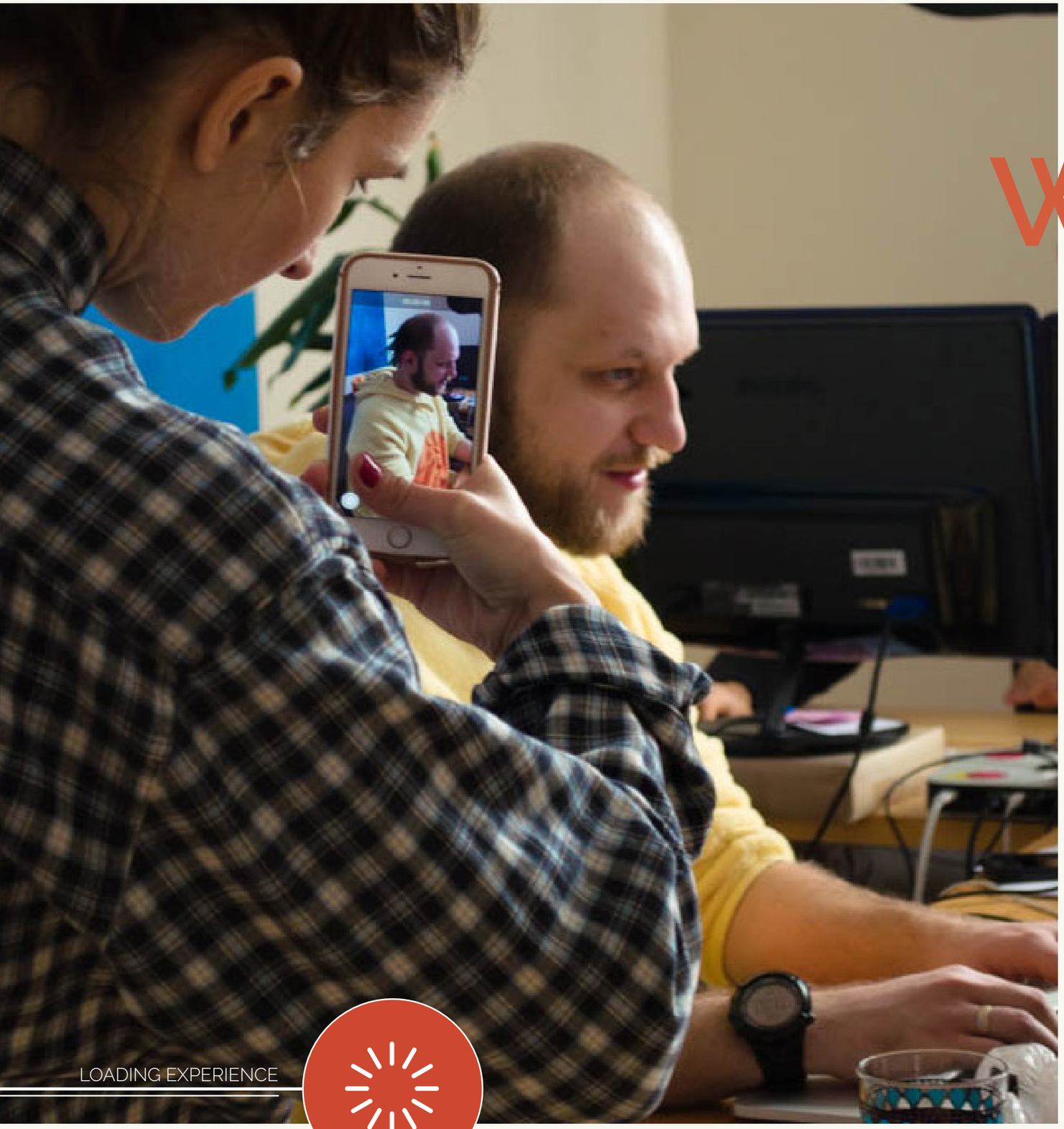
About TicketCo

TicketCo is creating the global standard for all kinds of event payments. Driven by a strong wish to make events easy, we gathered in 2011 to develop TicketCo – a self-service and user-friendly ticketing solution enabling organisers to rid themselves of expensive hardware and obsolete technology.

Since the launch in 2013, TicketCo has evolved into what it is today in collaboration with more than 1,600 event organisers. A partnership deal with iZettle in 2014 enabled an omnichannel solution by seamlessly integrating point of sales and making TicketCo even more complete.

By providing a frictionless payment journey for the public and a single point of administration for the event organiser, TicketCo is creating the best value for both users and organisers. Through TicketCo, organisers can sell both tickets, food and beverages, merchandise and accommodations.





LOADING EXPERIENCE

WORKFLOW

After many years we have learned some valuable lessons on having a set and effective work process.
This creates better projects and outcome for all parties.

Prototyping / usertesting
We are building clickable prototypes using InvisionApp / Marvel. We are utilising both internal user interview procedures for the sanity-check running interviews with the existing user groups.

A/B testing / funnelling / analytics
We are using tools like Hotjar (for identifying funnels in the product to reduce drop-offs) on the web and Appsee on mobile. We have extensive experience with Google Tag Manager / Google Analytics.

Prioritisation
Most common issue (and the main focus for the product manager) is prioritisation. We are utilising the combination of the RICE (reach, impact, confidence, effort) prioritisation score and the ProductBoard tool for working with the roadmap.

DevOps
We have solid knowledge with built tools, such as Git and Jenkins, and application containers, such as Docker and Kubernetes.

We also have experience with cloud computing services, such as Google Cloud, Microsoft Azure and Amazon Web Services.

We are building separate environments from the start. We have existing procedures for continuous integration (CI) and continues deployment (CD)
We are performing load and performance testing on most of the projects. We also use the tools for automated identification.

Infrastructure

Docker Containerization - containers are easily scalable, allowing to test software simpler and eases Infrastructure as Code (IaC) process. We use containers for both CI/CD processes and deployments.

Kubernetes - the de facto standard for container orchestration today. K8s is our main platform we use to run our workloads (both production and development). We also run our CI builds with this.

Google Cloud - provides Infrastructure, Platform, and Serverless computing environments. We use its Kubernetes Engine to run Kubernetes clusters. Also we use number of its Services/APIs.

AWS - We have experience using a number of its Services/APIs: EC2, ECS, Lambda, Elastic Beanstalk, S3, EFS, S3 Glacier, RDS, ElastiCache, CloudWatch, AWS Auto Scaling, Elasticsearch Service, CloudFront, Route 53, Rekognition, Cognito, GuardDuty, Inspector, WAF & Shield, SES.

Cloudflare - Integrated global cloud platform. We have experience using a number of its Services/APIs among them: Caching, Page Rules, Custom Pages, Workers etc.
Serverless computing - Cloud-computing execution model in which the cloud provider runs the server and dynamically manages the allocation of machine resources. We have experience using such runtimes as AWS Lambda and Cloudflare Workers.

Datastore

PostgreSQL - is a powerful, open-source object-relational database system with over 30 years of active development where it has earned a strong reputation for reliability, feature robustness, and performance.

MySQL - open-source bhrelational database management system.

Elasticsearch - is a distributed, RESTful search and analytics engine capable of solving a growing number of use cases. As the heart of the Elastic Stack, it centrally stores your data so you can discover the expected and uncover the unexpected.

Redis - is an open-source (BSD licensed), in-memory data structure store, used as a database, cache and message broker. It supports data structures such as strings, hashes, lists, sets, sorted sets with range queries, bitmaps, hyperloglogs, geospatial indexes with radius queries and streams.

InfluxDB - is used as a data store for any use case involving large amounts of time-stamped data, including DevOps monitoring, log data, application metrics, IoT sensor data, and real-time analytics.

Backend Programming Languages

Ruby - a dynamic, open source programming language with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to write.

Elixir - is a dynamic, functional language designed for building scalable and maintainable applications. Elixir leverages the Erlang VM, known for running low-latency, distributed and fault-tolerant systems, while also being successfully used in web development and the embedded software domain.

PHP - Popular general-purpose scripting language that is especially suited to web development. Fast, flexible and pragmatic, PHP powers everything from blog to the most popular websites in the world.

.NET - is a software framework developed by Microsoft that runs primarily on Microsoft Windows. It includes a large class library named as Framework Class Library (FCL) and provides language interoperability across several programming languages. Programs written for .NET Framework execute in a software environment named the Common Language Runtime (CLR). The CLR is an application virtual machine that provides services such as security, memory management, and exception handling. FCL and CLR together constitute the .NET Framework.

Frontend and Stylesheet Languages

JavaScript — a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm. Alongside HTML and CSS, JavaScript is one of the three core technologies of the World Wide Web.

Sass — is the most mature, stable, and powerful professional grade CSS extension language in the world. It is completely compatible with all versions of CSS.

Frontend Library and Module Bundler

React.JS - a JavaScript library for building user interfaces. It is maintained by Facebook and a community of individual developers and companies. React can be used as a base in the development of a single-page or mobile applications. Complex React applications usually require the use of additional libraries for state management, routing, and interaction with an API.

Webpack - open-source JavaScript module bundler primarily for JavaScript, but it's also used to transform front-end assets like HTML, CSS. Webpack takes modules with dependencies and generates static assets representing those modules.

Native Mobile Development

Swift - is a general-purpose programming language built using a modern approach to safety, performance, and software design patterns. The goal of the Swift project is to create the best available language for uses ranging from systems programming, to mobile and desktop apps, scaling up to cloud services. Most importantly, Swift is designed to make writing and maintaining correct programs easier for the developer.

Kotlin - is a statically typed programming language that runs on the Java virtual machine and also can be compiled to JavaScript source code or use the LLVM compiler infrastructure.

Content Management System

WordPress - content management system based on PHP and MySQL. WordPress is most associated with blogging (its original purpose when first created) but has evolved to support other types of web content including more traditional mailing lists and forums, media galleries, membership sites, learning management systems (LMS) and online stores. WordPress is used by more than 60 million websites, including 33.6% of the top 10 million websites as of April 2019, WordPress is one of the most popular CMS solutions.

Web Frameworks

Ruby on Rails, or Rails - is a server-side web application framework written in Ruby. Rails is a model–view–controller (MVC) framework, providing default structures for a database, a web service, and web pages. It encourages and facilitates the use of web standards such as JSON or XML for data transfer, HTML, CSS and JavaScript for user interfacing.

Phoenix Framework - is a web development framework written in the functional programming language Elixir. Phoenix uses a server-side model-view-controller (MVC) pattern. Based on the Plug library and ultimately the Cowboy Erlang framework, it was developed to provide highly performant and scalable web applications.

Erlang OTP - is a programming language and runtime system for building massively scalable soft real-time systems with requirements on high availability. OTP is a set of Erlang libraries, which consists of the Erlang runtime system, a number of ready-to-use components mainly written in Erlang, and a set of design principles for Erlang programs.

Apollo Client - is the best way to use GraphQL to build client applications. The client is designed to help you quickly build a UI that fetches data with GraphQL and can be used with any JavaScript frontend.

What is the most important thing in collaboration? We think it's trust. How do you build trust? Through transparency, involvement and integrity. Each new initiative or common project we start with a face to face meeting and a workshop.

We want to hear your story, understand the goal, but also to find out if we are able to work together. We know from experience that it's much better to find out early in the process if there are communication issues, differences in values, or just incompatible.

Standards

GDPR - regulations in EU law on data protection and privacy for all individuals within the European Union and the European Economic Area.
REST API - a software architectural style that defines a set of constraints to be used for creating web services. Web services that conform to the REST architectural style provides interoperability between computer systems on the Internet.
GraphQL - an open-source data query and manipulation language, and a runtime for fulfilling queries with existing data.
SOLID / DRY - principles of software development that allows a programmer to create a system to easily maintain and extend over time.
International Software Testing Qualifications Board (ISTQB) - a standardized qualification for software testers.
UX - the process of enhancing user satisfaction with a product by improving the usability, accessibility, and pleasure provided in the interaction with the product.
A11Y - we strive to develop our products most accessible by each person, meaning we code them in a way to fully comply with WAI-ARIA specifications and support all kinds of screen readers and multiple navigation methods.

Quality Assurance

Activities:

- Analyze requirements
- Design test flows
- Perform testing on all stages of development
- Create documentation (end-to-end functional scenarios or a check list of the final verification (Acceptance testing))
- Report issues and revalidate fixes

Main QA phases:
Phase #1

- Functional testing is mainly to confirm that the functionalities match the requirements.
- Confirmation testing is performed to validate corrected functionalities made after the client's feedback or defects found in the system.
- Integration testing is to validate if features work together.

Phase #2

- Regression testing ensures new functionality doesn't introduce new defects into a product.

Phase #3

- System testing is used to provide acceptance of the system.

Approaches

Agile - iterative approach to project management and software development that advocates adaptive planning, evolutionary development, early delivery, and continual improvement as well as rapid and flexible response to change.
Test-driven development - a software development process that relies on the repetition of a very short development cycle. The requirements are turned into very specific test cases and then the software is improved to pass the new tests.
Code Review - a software quality assurance activity in which one or several persons check a program mainly by viewing and reading parts of its source code. They do so after implementation or as an interruption of implementation.
Code linting - the process of running a program that analyse code for potential errors.
Code Security Check - the process of running a program that detects various security vulnerability patterns, and tainted analysis to track user input data, etc.
CI/CD - refers to the combined practices of continuous integration and continuous delivery. We use automated pipelines for both automated testing and deployments. We usually have a set of environments (Production/Staging/Development) to move code from one environment to another, such as Dev to Stage or from Stage to Prod, while we run automated and manual tests and checks on them.
Infrastructure as Code - managing and provisioning computer data centers through machine-readable definition files rather than physical hardware configuration or interactive configuration tools.

Zero-waste is part of our culture at Innocode. We apply this philosophy both physically and operationally. In the physical sense, we sort waste at our Lviv office by having 12 different bins for different types of waste and installed separate compost bins for the organic waste. We also hold educational meet-ups for other companies in the area.
In the operational sense, we consider time and effort as a resource that should be used wisely. These have an impact on our working processes and we apply them through lean methodologies and verifying hypotheses before actual implementation.

ZERO WASTE

Digital Products Development

- together we create relevance

Innocode Norway

Morten Holst
+47 934 25 000
morten@innocode.no

Valerii Shypunov
+38 066 700 49 99
valerii@innocode.no

Innocode Sweden

Tomas Falk
tomas@innocode.com
+46 736 63 6790

Bjarte Falck Olsen
bjarte@innocode.com
+ 46 763 98 2109

Innocode U.S.

Steinar Bjørnsen
steinar@innocode.com
+1 415 351 8515