

K2 Platform

Create, don't start from scratch

A Ukrainian technology platform for creating, modernizing, and evolving business applications.

Python · TypeScript · Vue · PostgreSQL · ER/BP · ORM · API

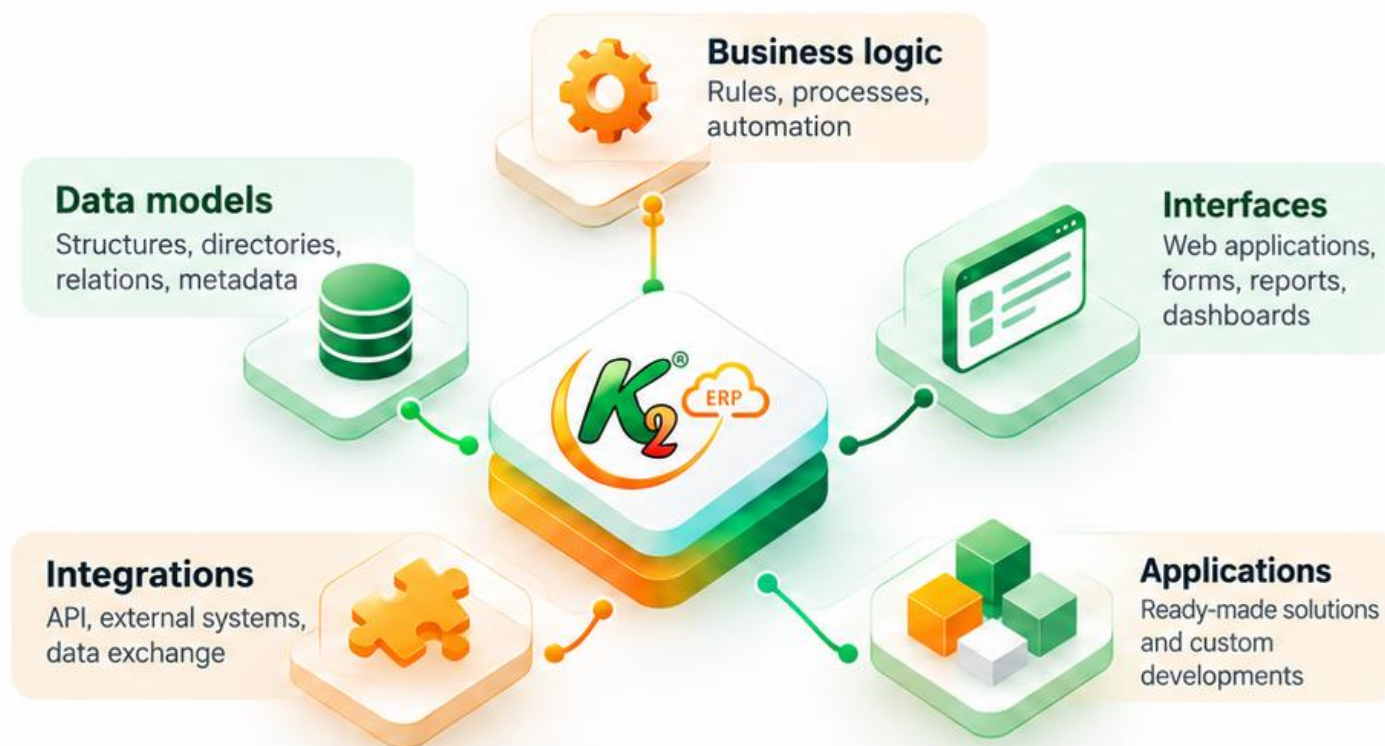


Advantage:

ready-made core development mechanisms on a single platform.



Not every team needs to reinvent its own wheel of forms, directories, and reports.



K2 – Ukrainian Technology Company

with a 30+ year history of software development



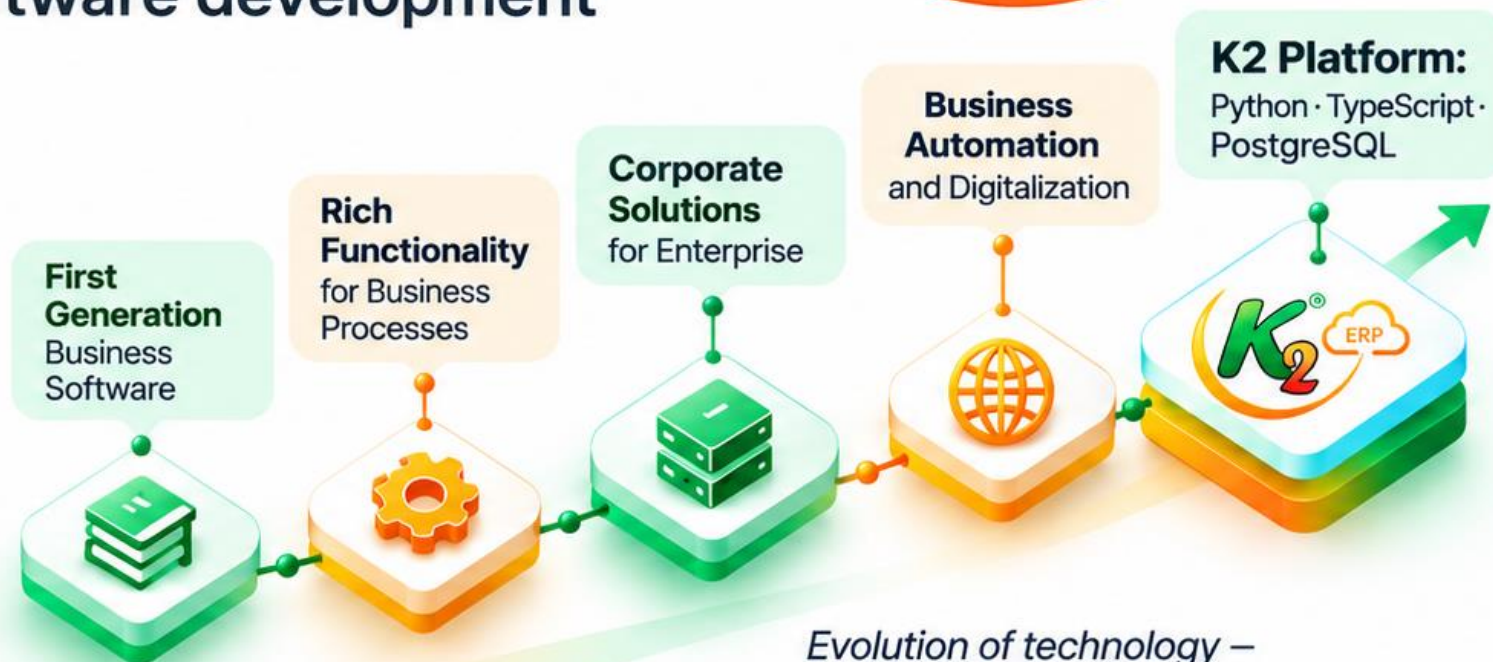
Over 30 years of experience
in developing custom software.



From first-generation systems
to modern development tools
for today's technological platforms
such as Python, TypeScript and PostgreSQL.



Created in Ukraine.
Growing for the international market.



*Evolution of technology –
the key to yesterday, today and tomorrow.*



A confident step into the future
with the K2 platform.

We know old technologies, because we created the programs ourselves



Clipper, Basic, Delphi, Pascal, SQL, PHP, Python, TypeScript.

We have gone a long way from desktop systems to web architecture, from manual form programming to data models and development automation.

We know not only how to create new programs, but also how the old ones were built.

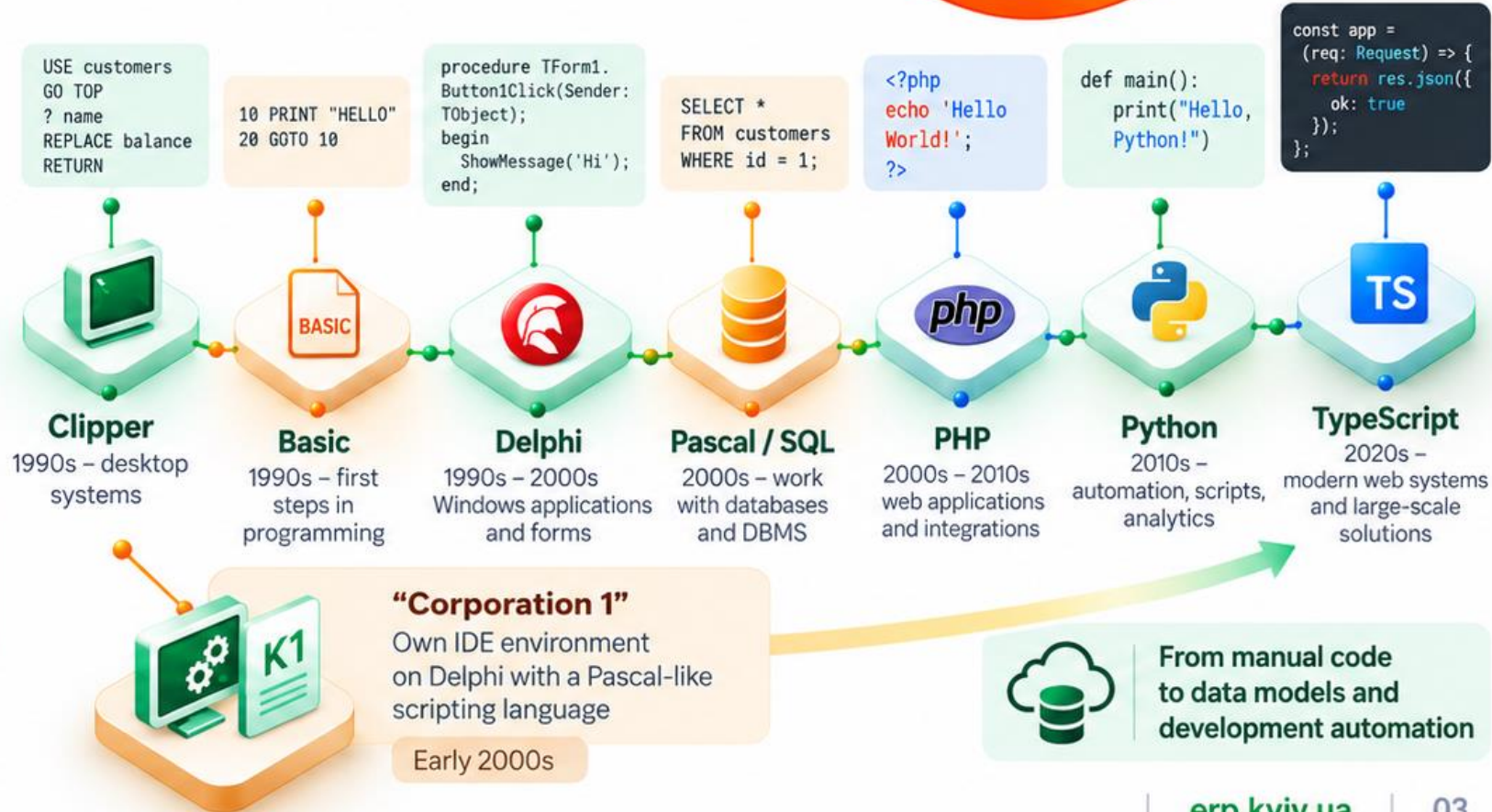


Our experience is your advantage

We understand the legacy, help to develop and integrate it.



Old technologies are not a problem — they become a problem only when left behind.



From custom programs – to a universal platform



Over the years, we have developed a large number of business solutions: accounting, warehouse, production, CRM, maintenance (TOIR), document management, e-learning, analytics and more.

For further growth, we created and continue to improve our own technological tools.

Today **K2** is a platform on which not only our products, but also your software solutions can be built.



The advantage:

A single platform can be the foundation for many practical solutions and new products.



*When the modules and tools are diverse —
it is not a collection of technological contradictions,
but a unified platform.*



Ukrainian Technologies for the Global Market



Ukrainian development. A modern international technology stack.

Python, TypeScript, Vue, PostgreSQL, web architecture, accessible source code, API, and integration capabilities.

Technological independence begins with the ability to develop your own software.



Advantage:

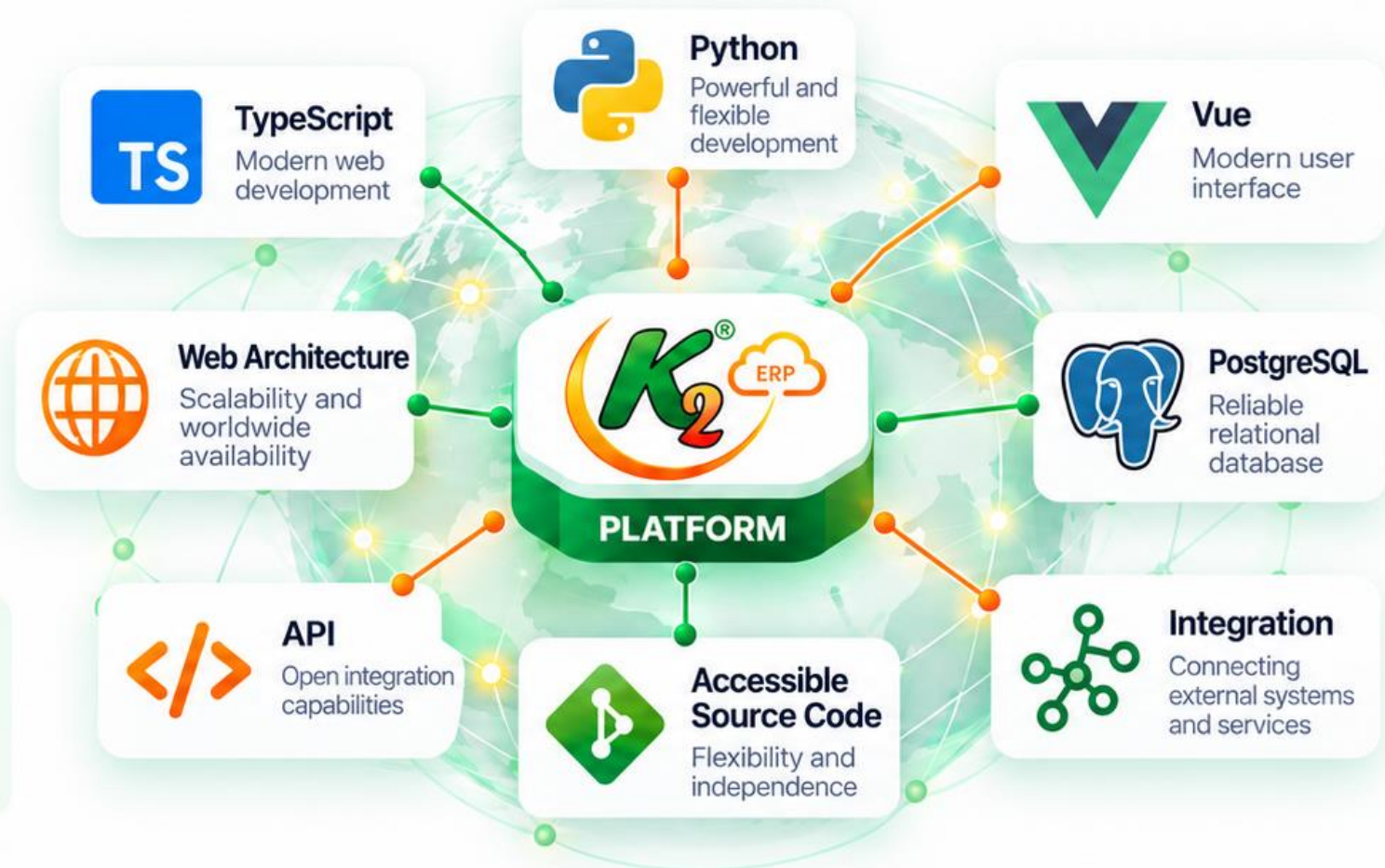
Widely used technologies and a proprietary platform provide greater control over the development of software products.



*The global market loves modern technologies.
Even more — when you truly master them.*



Developed in Ukraine



K2 Platform

A technological platform for creating, modernizing and developing business applications.

Python · TypeScript · Vue · PostgreSQL · ER/BP · ORM · API



Advantage:

ready-to-use basic development mechanisms on a single platform.



Business logic

Processes, rules, automation



Integrations

Connection to external systems and services



Data models

Structure, relationships, rules



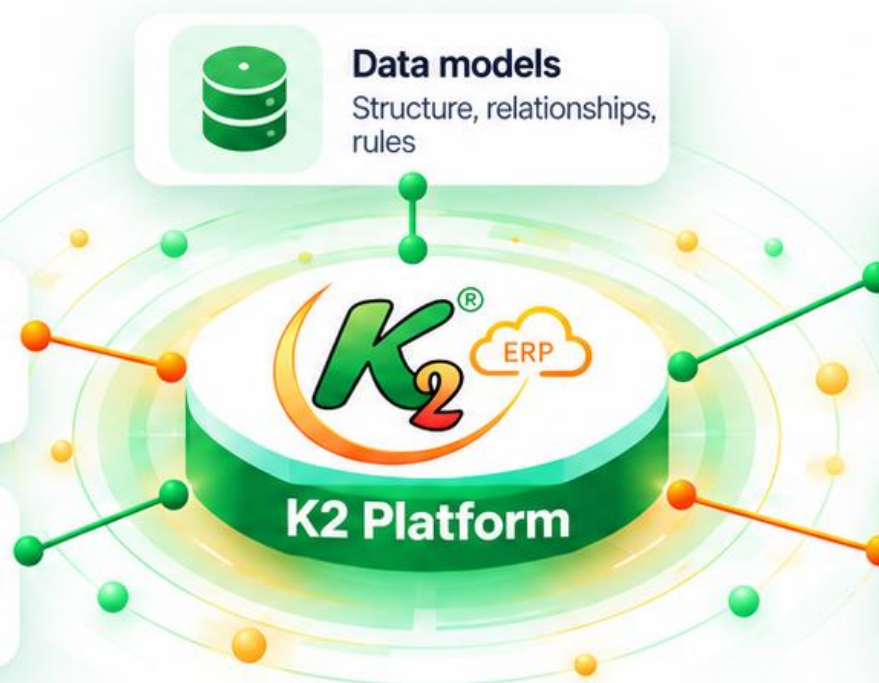
Interfaces

Modern user experience



Ready-to-use applications

Fast deployment and scaling



Legacy programs do not necessarily have to stay in the past



Your 20-year-old programs?

They can be an advantage.

They already contain business logic, proven algorithms and domain knowledge. The problem is often not what they do, but the technologies on which they run.

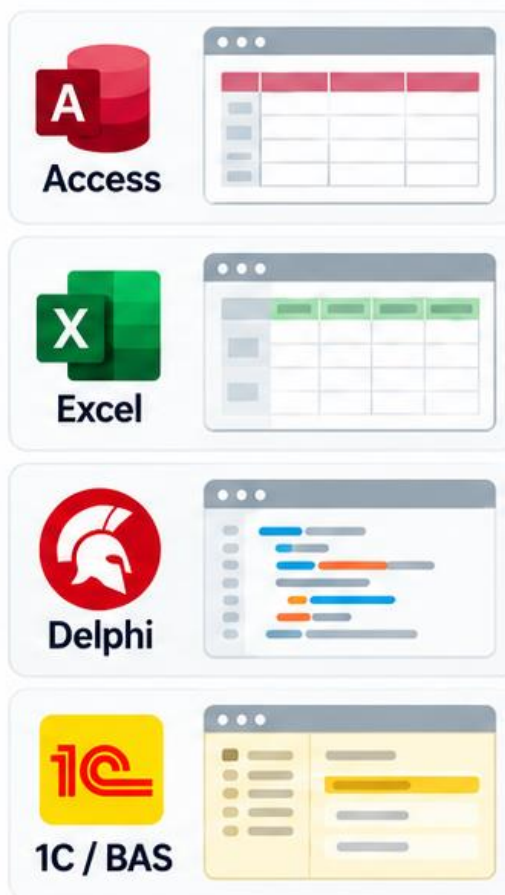


Advantage:

preserving useful functionality during architecture modernization.



Legacy programs often work longer and more reliably than many modern technologies. And we should not get rid of them.



Create a new product, not another bicycle



Why should every team build forms, directories, ORM, access rights, reports and update systems again from scratch?

K2 provides ready-made mechanisms for building your own business solutions.



Advantage:
less duplication
of infrastructure code.



*A bicycle is a great invention.
But you don't have to assemble one every time from scratch.*



What does the K2 Platform include?



- **Modeling:**

ER, BP, YAML.

- **Development:**

Python, ORM, PostgreSQL.

- **Interfaces:**

TypeScript, Vue, ready-made components.

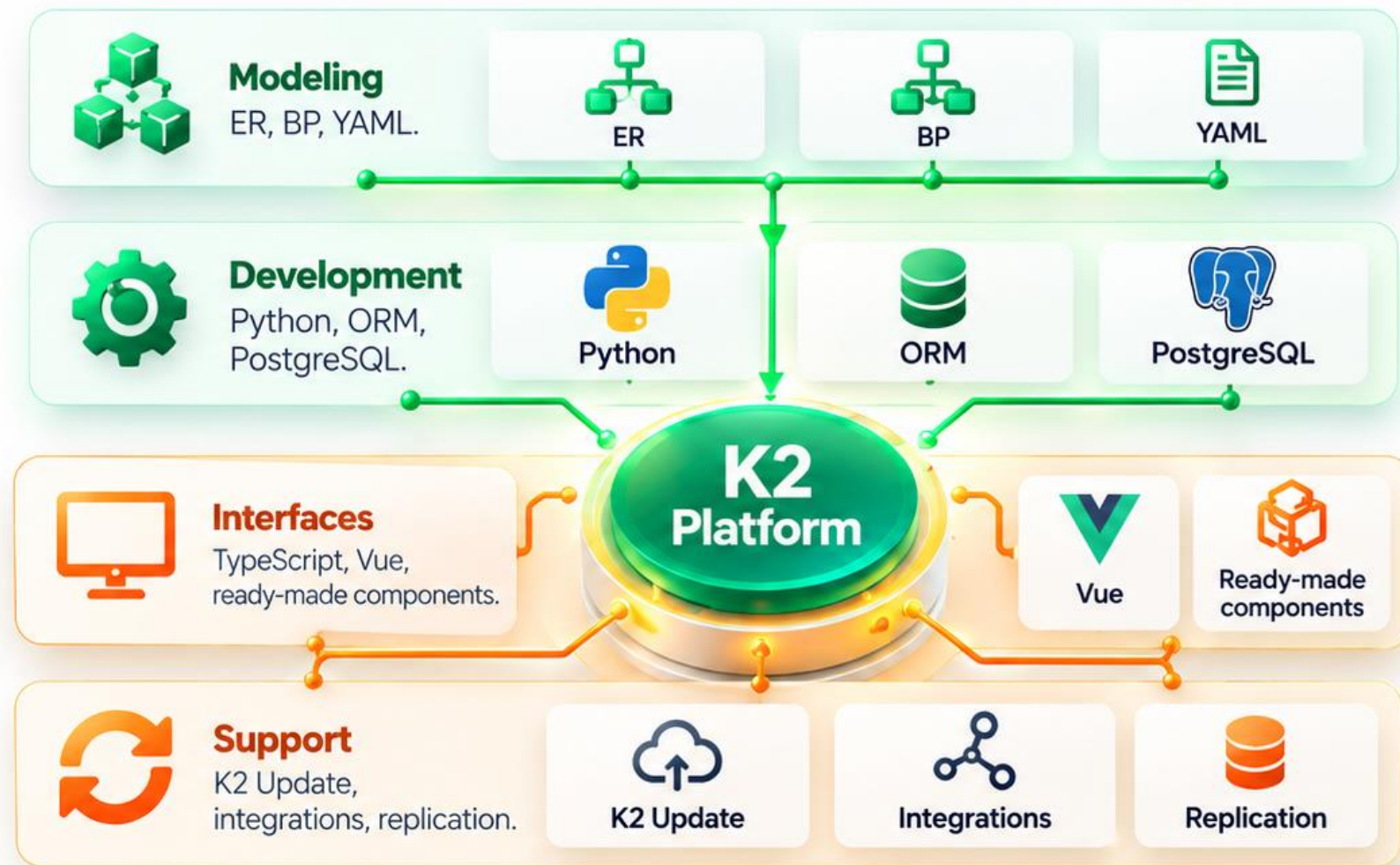
- **Support:**

K2 Update, integrations, replication.



Advantage:

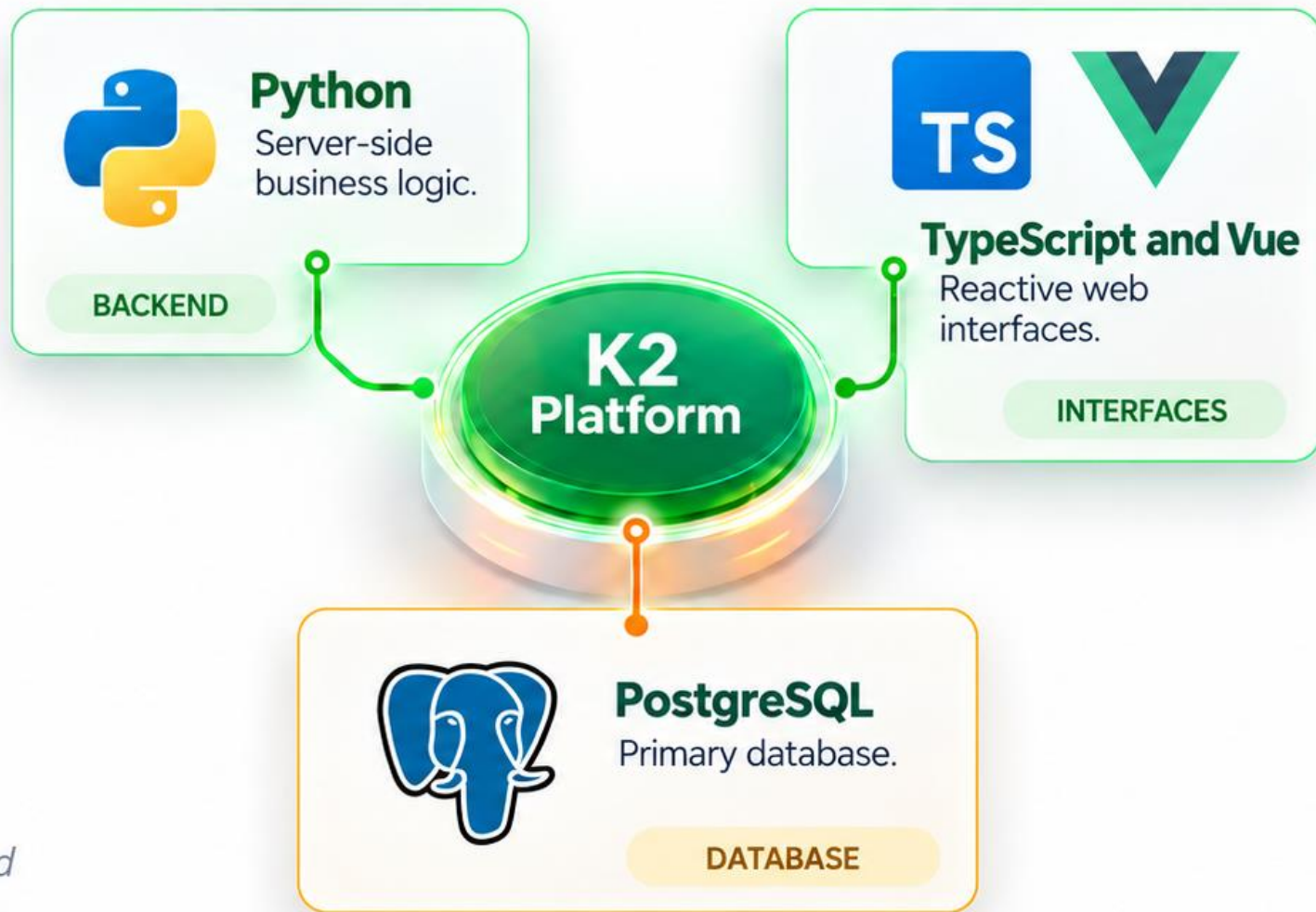
Interaction of core mechanisms in a unified technological environment.



Technology Stack Without Exotic Tools



- **Python** — server-side business logic.
- **TypeScript and Vue** — reactive web interfaces.
- **PostgreSQL** — primary database.
- **Knowledge of these technologies** remains useful far beyond K2.



Advantage:

the use of widespread languages and technologies.

Good technology is the kind that can be understood by more than just one developer who is on vacation.



Open source code instead of a black box



You can read, analyze, modify and develop the code according to your license terms.

Python, TypeScript, YAML, Git and other standard developer tools.



Advantage:

transparency of software implementation and the ability to develop your own solutions.



A developer doesn't need yet another special IDE



Work with code using familiar tools.

Git, Python editors, development tools, TypeScript and testing tools can all be part of your K2 development process.



Advantage:

the ability to use popular developer tools.



A special IDE is just as exciting as the first day of a new developer. But in a corporate system — not always.



One platform — many application classes

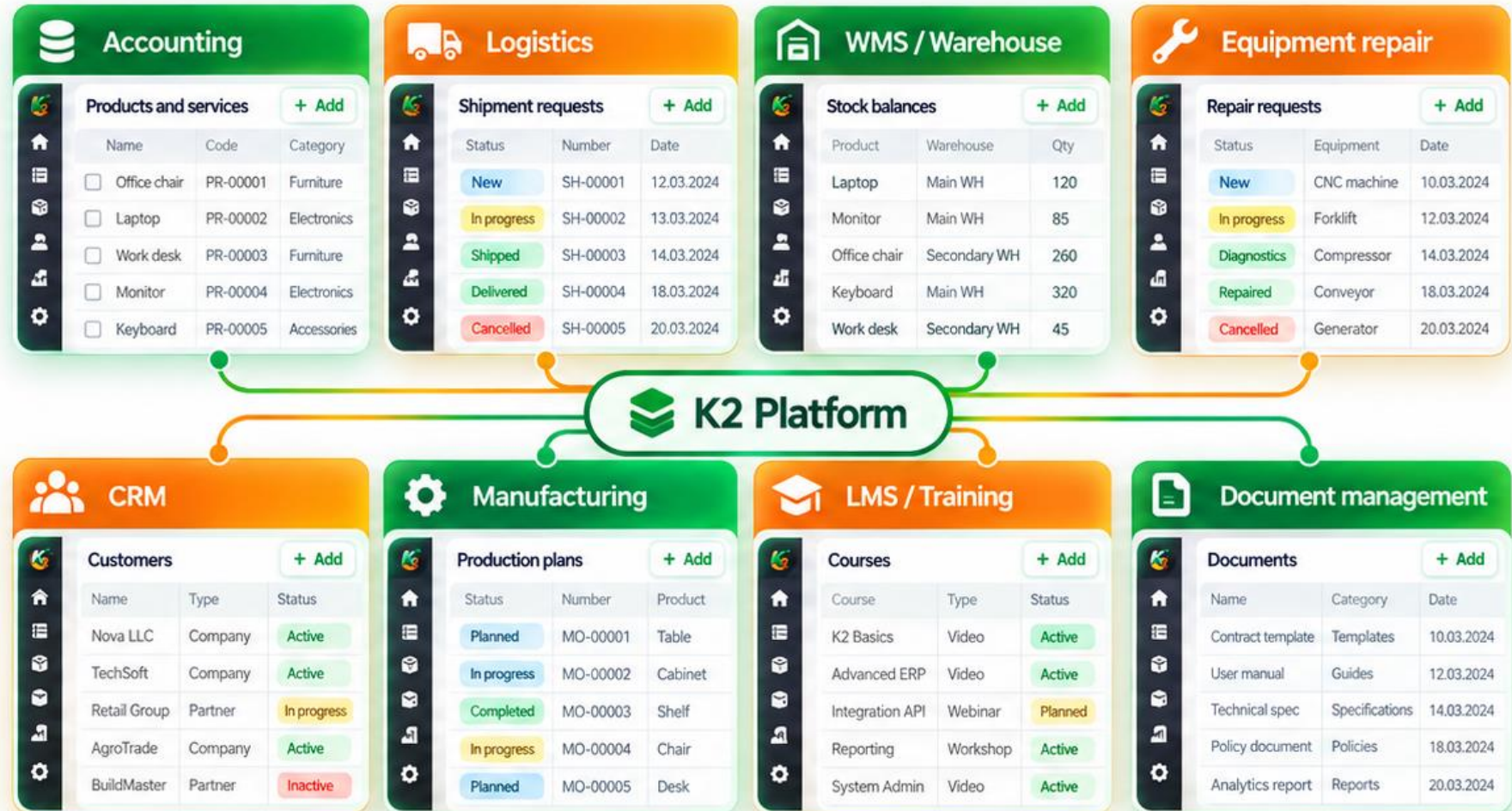


K2 enables solutions for accounting, logistics, warehousing, equipment repair, CRM, manufacturing, training, document management and analytics.

Different tasks.
A single technological foundation.



Advantage:
a wide range of components and approaches.

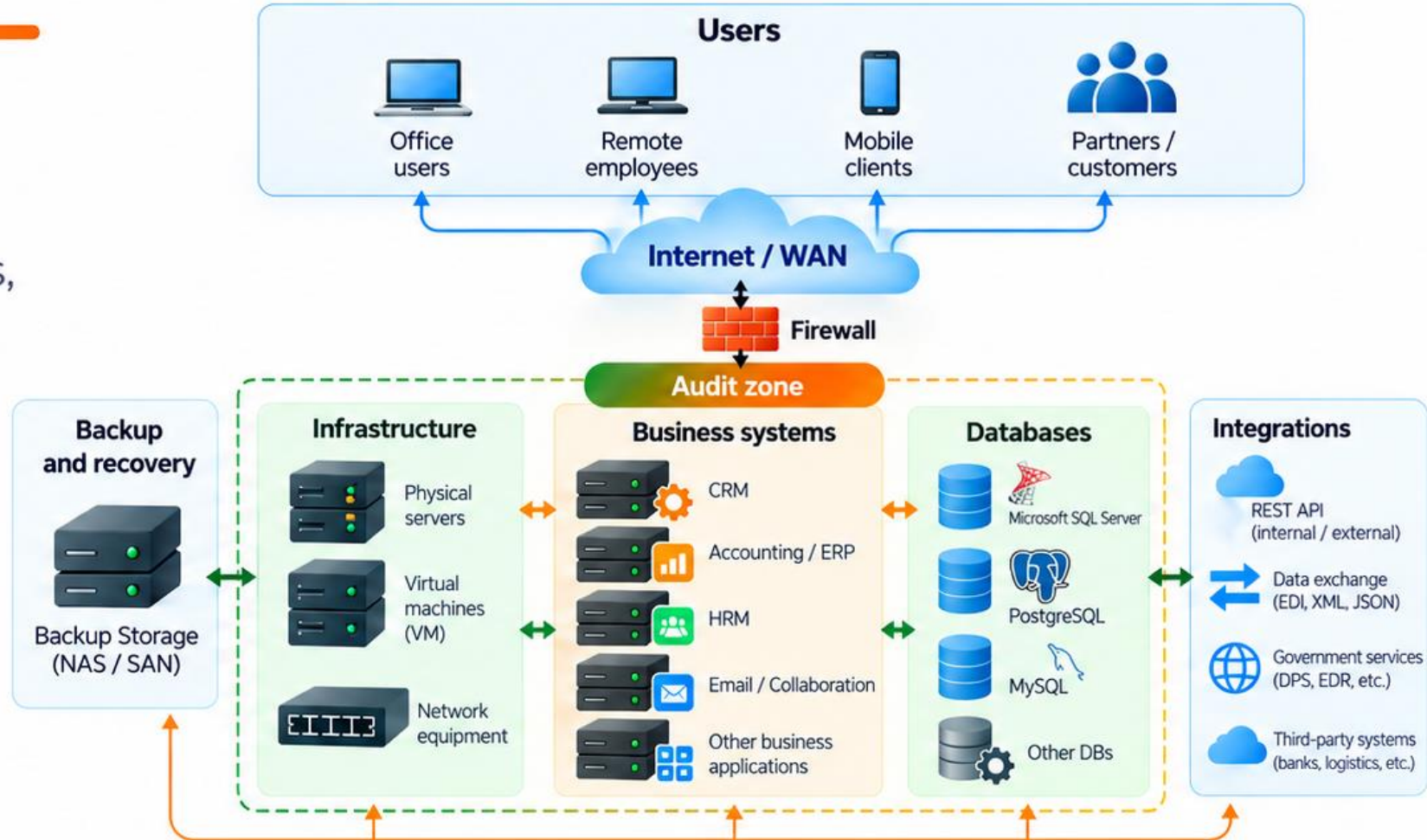


Before you modernize — you need to understand



IT infrastructure and software audit.

We analyze systems, dependencies, data, integrations, technical risks and modernization opportunities.



Advantage:

a well-grounded
modernization plan
instead of trial and error.

Audit of servers, virtualization and networks



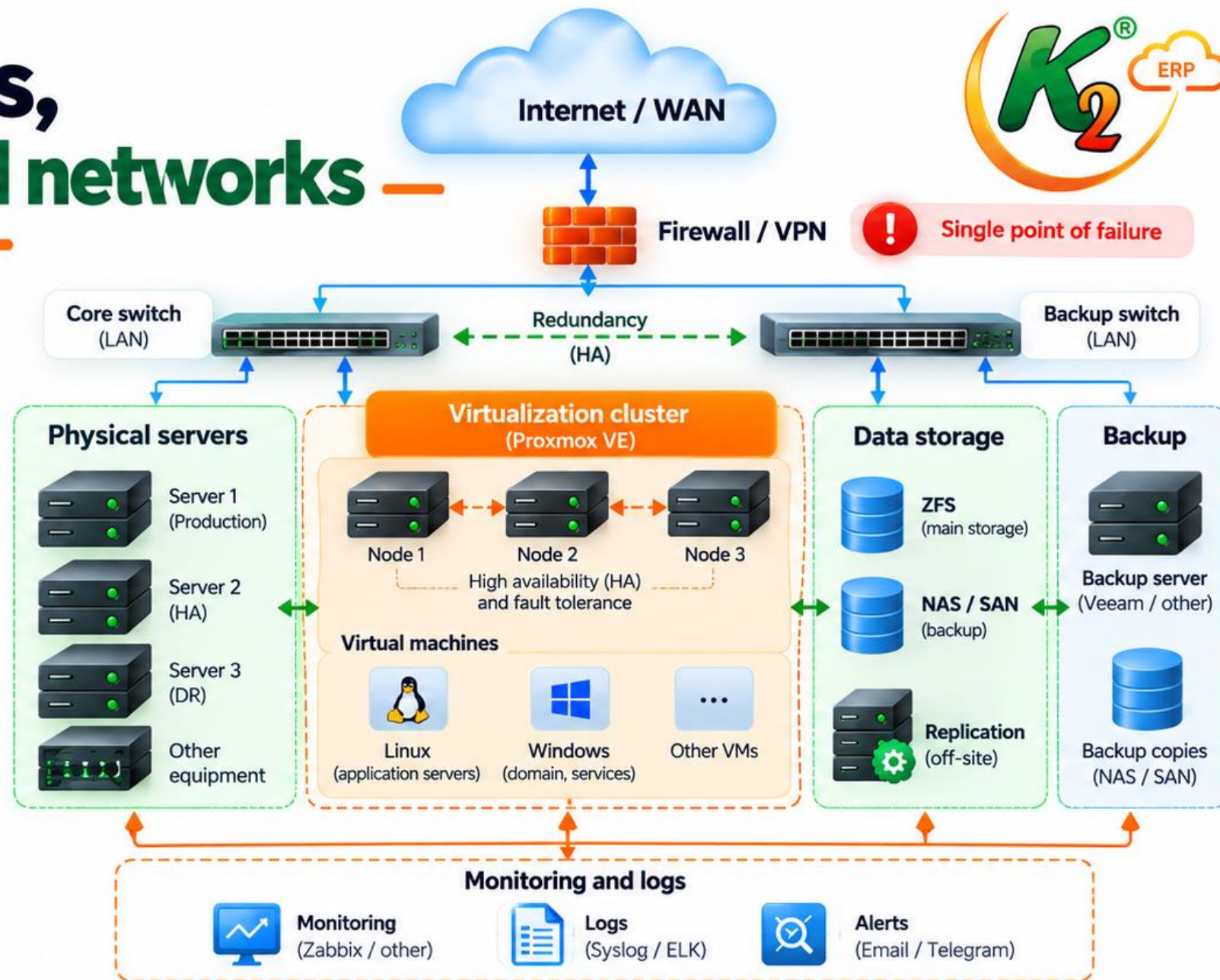
We analyze server hardware, virtual machines, operating systems, storage, network architecture, fault tolerance and backup.

We identify critical dependencies and single points of failure.



Advantage:

the audit takes into account the real production environment of your software.



Audit of the software landscape



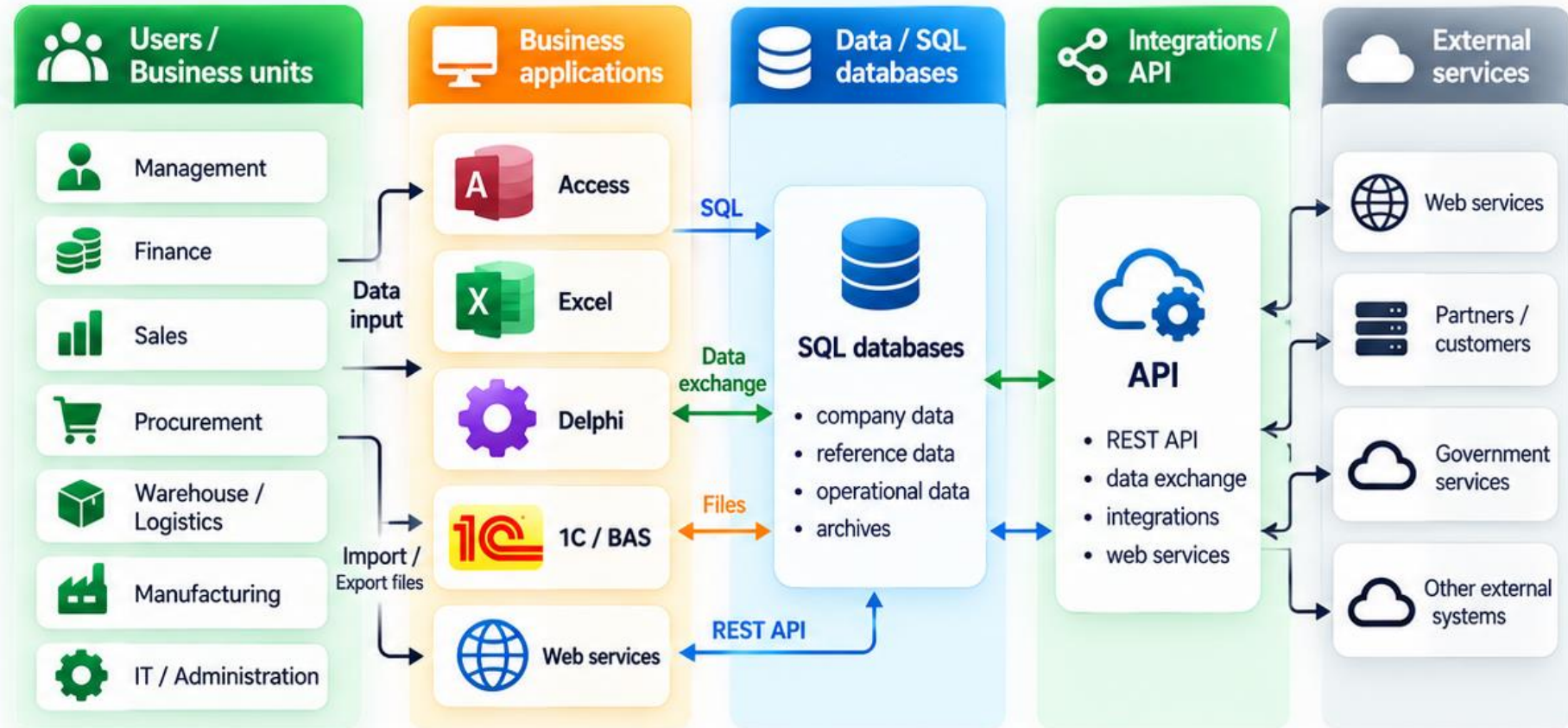
What software is used?
Where does the data come from?
What APIs are being used?
Who supports the code?
What licenses apply?

Result — a map of software systems and their integrations.



Advantage:

understanding the cost of support and the potential for system replacement.



The more software a company has, the more documentation an audit makes especially valuable.

Audit of security and business continuity



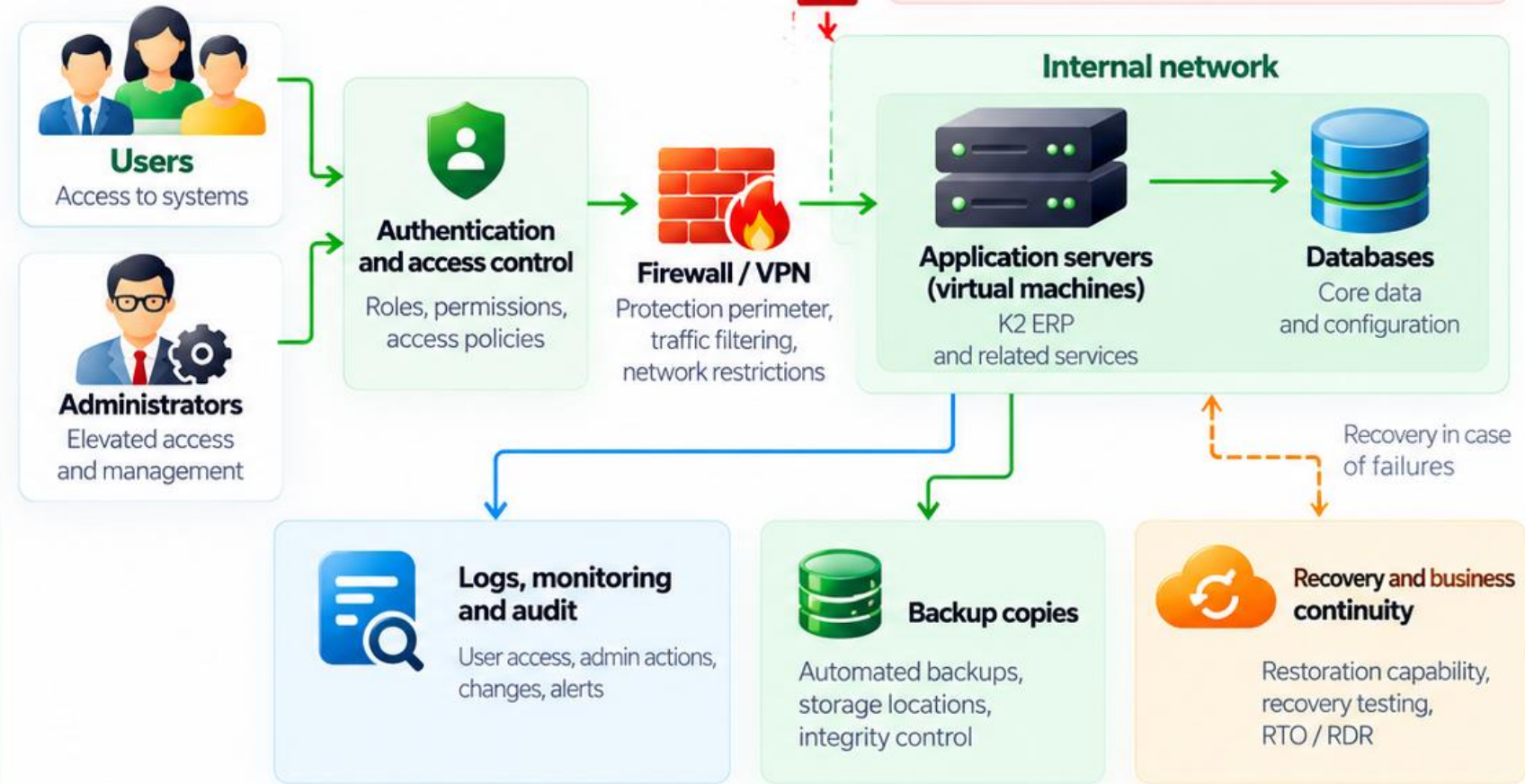
User rights, administrative access, logs, network restrictions, backups, recovery capabilities and risks of third-party components.

A backup is not yet a recovery guarantee.



Advantage:

identification of technical risks before migrating to a new system.



Audit of the source code — and technical debt



We analyze the code, module structure, dependencies, SQL queries, outdated libraries, duplicated logic and the complexity of the codebase.

Python

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.orm import sessionmaker
3
4 engine = create_engine(
5     "postgresql://user:***@db:5432/erp",
6     pool_size=10,
7     max_overflow=20,
8     pool_pre_ping=True
9 )
10 Session = sessionmaker(bind=engine)
```

TS TypeScript

```
1 import { ref, computed } from 'vue';
2 import axios from 'axios';
3
4 export async function loadData(id: number) {
5     const response = await axios.get(
6         '/api/documents/${id}'
7     );
8     return response.data;
9 }
```

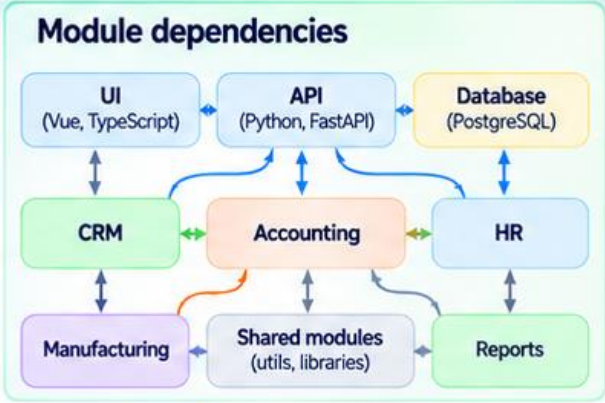
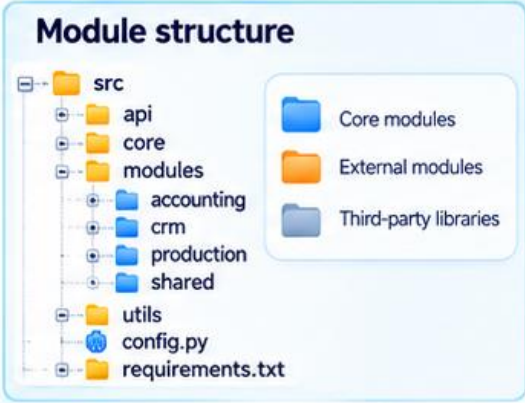
Delphi

```
1 procedure TMainForm.Button1Click(
2     Sender: TObject);
3 var
4     Q: TFDQuery;
5 begin
6     Q := TFDQuery.Create(nil);
7     try
8         Q.SQL.Text := 'SELECT * FROM ORDERS
9             WHERE ID = :ID';
10        Q.Params[0].AsInteger := EditID;
11        Q.Open;
12        finally
13            Q.Free;
14        end;
```

SQL

```
1 SELECT o.id,
2        o.number,
3        c.name AS customer_name,
4        SUM(i.amount) AS total
5 FROM orders o
6 JOIN customers c ON c.id = o.customer_id
7 LEFT JOIN order_items i ON i.order_id = o.id
8 WHERE o.date >= '2024-01-01'
9 AND o.status = 'ACTIVE'
10 GROUP BY o.id, o.number, c.name
11 ORDER BY o.date DESC;
```

We identify what can be preserved, what needs to be refactored, and what should be replaced — and provide recommendations.



Analysis results matrix

Component / module	Status	Main issue	Recommendation
UI (Vue 2)	!	Outdated version, security vulnerabilities	Update to Vue 3
Backend (Python 3.8)	!	End of support, security risks	Upgrade, refactor critical parts
Delphi modules	!	Complex logic, legacy components	Optimize, replace where possible
SQL queries	!	Inefficient queries, lack of indexes	Optimize, add indexes
Third-party libraries	!	Outdated versions, known vulnerabilities	Update or replace
Duplicated logic	!	Code duplication in several modules	Extract to shared modules
Tests	!	Missing or insufficient tests	Add automated tests



Advantage:

focus only on necessary and proven improvements and replacement mechanisms.



Identify



Preserve



- Keep
- Refactor
- Replace



Experience working with different systems and technologies



Experience working with Clipper, Delphi, Basic, 1C/BAS, PHP, Python, JavaScript/TypeScript, SQL, web technologies and corporate infrastructure.

We understand not only modern tech stacks, but also legacy systems that the business is moving from.



Advantage:

technological versatility and experience in migration between generations of software.



The result of the IT audit — a roadmap for modernization



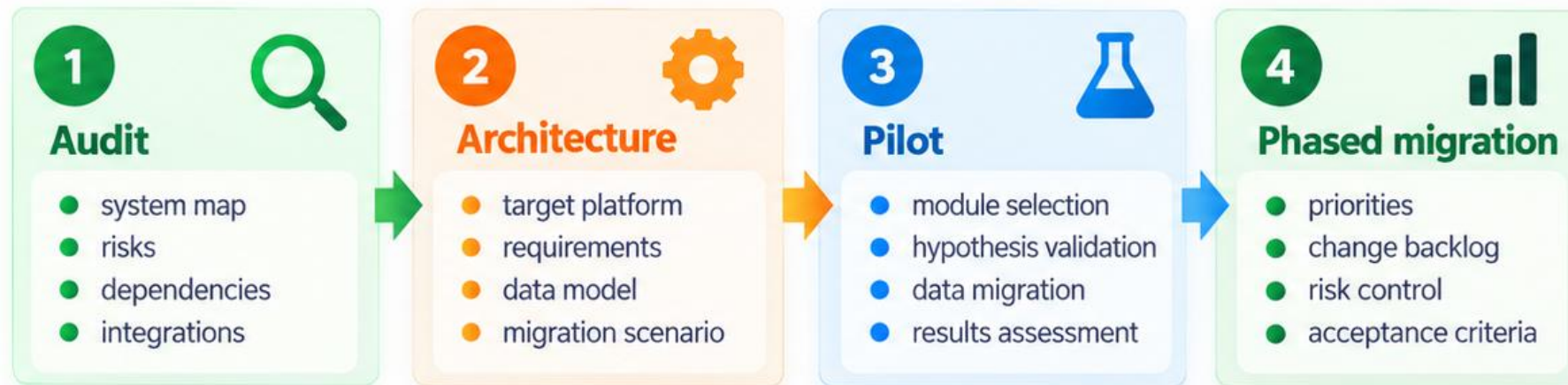
We obtain a map of systems, a list of risks, integration schemes, requirements for a new platform, migration backlog and acceptance criteria.

Not every program stage needs to be done first.



Advantage:

ability to plan the transition without unjustified business disruptions.



Logic of determining priorities



K2 — platform and engineering competence



We can not only deploy a new technology platform, but also leverage our experience with existing systems, improve the architecture, migrate data and develop new solutions.

**Technology matters.
Understanding the task
matters even more.**



Advantage:

a comprehensive approach from assessment of existing systems to building and launching a new product.



Reverse engineering – to preserve business logic



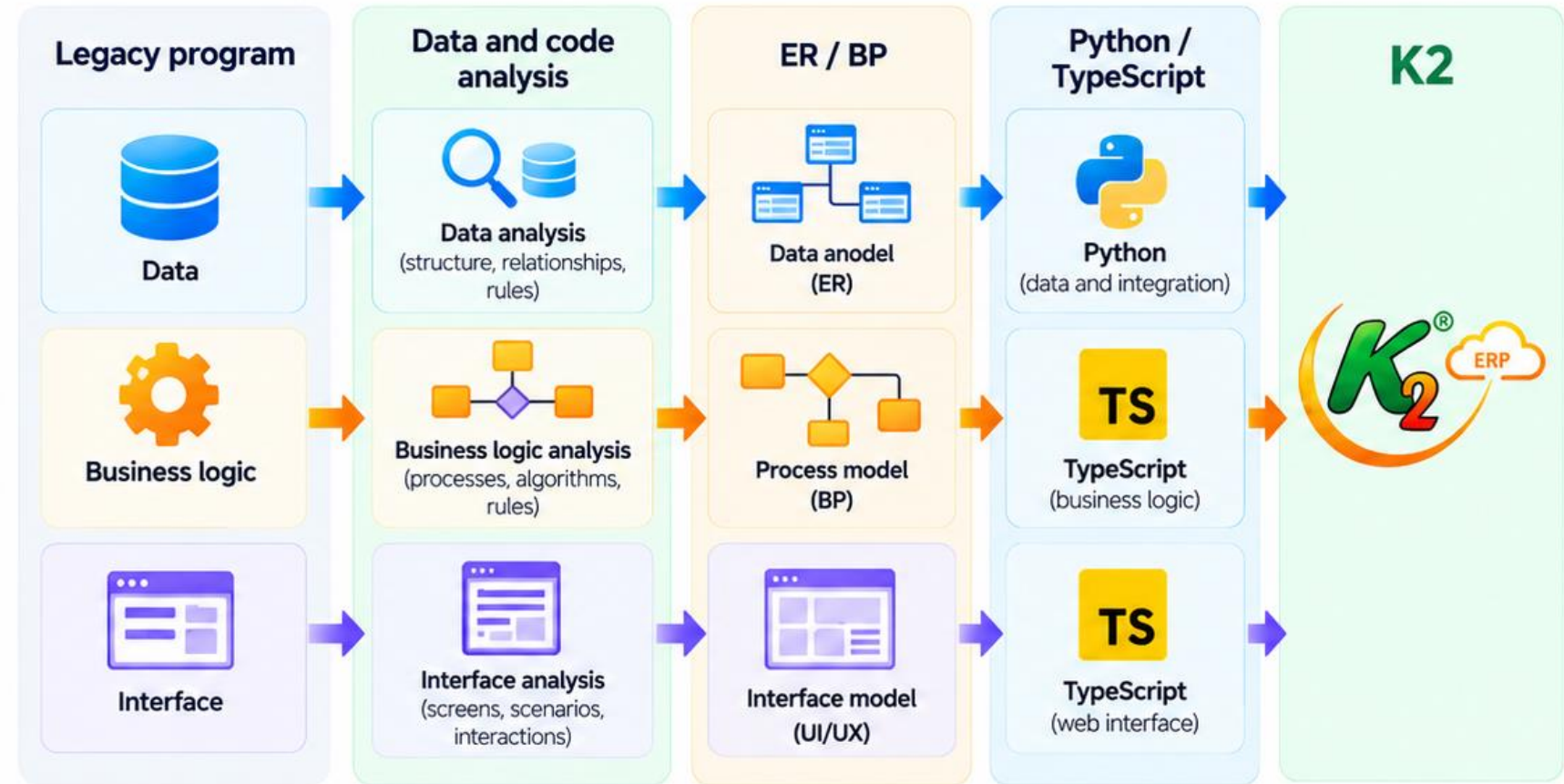
Old programs contain years of work, thousands of business rules and domain knowledge.

Our goal is to study them, structure the logic and recreate it on a modern platform.



Advantage:

preservation of investments in accumulated functionality.



Five levels of reverse engineering



- 1 Data analysis.
- 2 Model restoration.
- 3 Business logic analysis.
- 4 Recreating functionality.
- 5 Validation and implementation.

Modernization is successful when the transition from analysis to implementation happens consistently, with verification at each level.



Advantage:

a controlled, verified modernization process.

1



Data analysis

Result: structure of DB, files, tables, queries and data sources.

2



Model restoration

Result: ER model, relationships, entities, attributes and structure rules.

3



Business logic analysis

Result: algorithms, calculations, validations, processes and workflow scenarios.

4



Recreating functionality

Result: new implementation in Python, TypeScript, ORM, API and web components.

5



Validation and implementation

Result: testing, result verification, data migration and production rollout.

What exactly we transfer from the old system



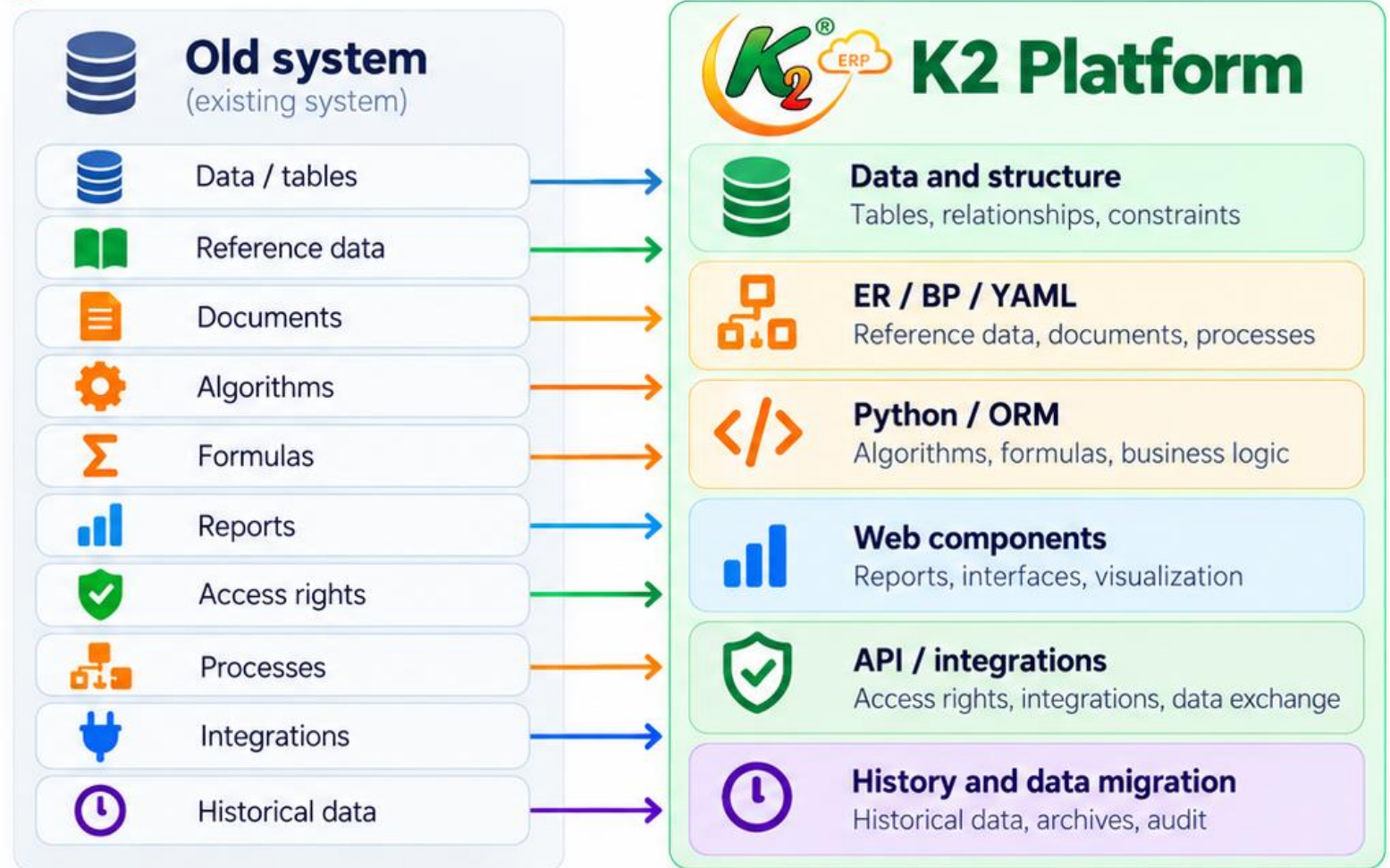
Not only tables.

Reference data, documents, algorithms, formulas, reports, access rights, processes, integrations and historical data.



Advantage:

attention to the functional completeness of the transition.



Microsoft Access: from a local database to a web solution



Tables, queries, forms, reports and VBA are analyzed separately. The data structure is transferred to PostgreSQL, business logic and rules are rewritten, and a new user interface is created in K2. After a full modernization, users no longer need the old Access application.



Microsoft Access:

Forms and queries are also business logic



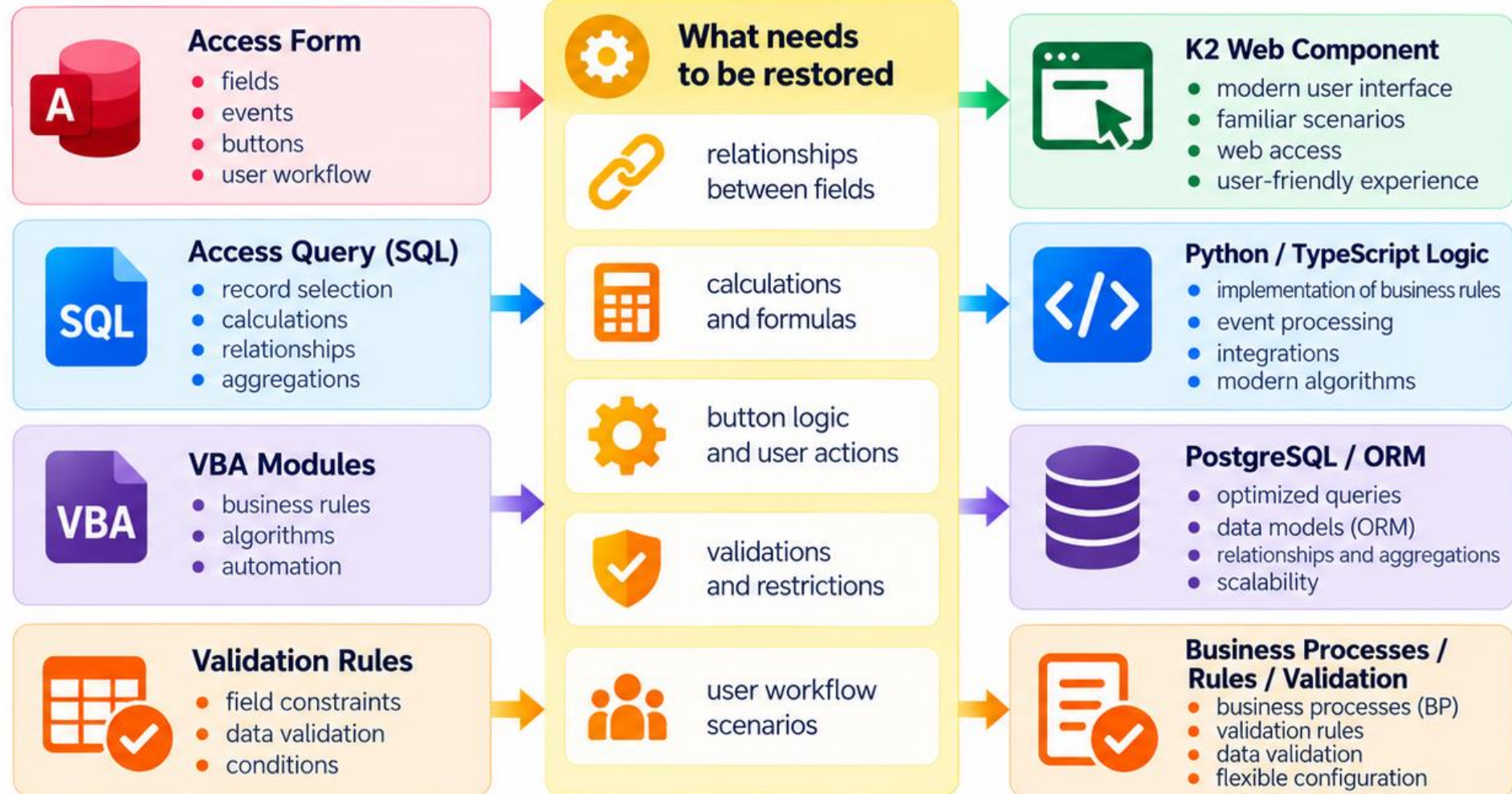
Transferring a table is easy. Much more important is to restore dependencies, validations, calculations and user actions — because this is exactly where the business logic resides.

Forms and queries implement this logic in the database program.



Advantage:

functionally equivalent modernization, not just a formal migration.



Excel: when a spreadsheet has long become a system



Complex formulas, VBA macros, links between sheets, combined tables and manual processes.
Excel can be a useful tool. But not every information system should remain a file.

Sales_Analysis_v2024.xlsx - Excel

File Home Insert Page Layout Formulas Data Review View Developer Tell me what you want to do...

Clipboard Font Styles

F2 =IF(E2="Completed",SUM(Orders!\$G:\$G),0)

ID	Product	Category	Quantity	Unit Price	Amount	VAT	Total	Status
1001	Laptop	Electronics	5	1,200.00	6,000.00	1,200.00	7,200.00	Completed
1002	Monitor	Electronics	10	450.00	4,500.00	900.00	5,400.00	In progress
1003	Keyboard	Accessories	25	85.00	2,125.00	425.00	2,550.00	Completed
1004	Mouse	Accessories	30	25.00	750.00	150.00	900.00	In progress
1005	Office Chair	Furniture	8	320.00	2,560.00	512.00	3,072.00	Completed

Sales by Category

Category	Amount	Orders
Electronics	10,500.00	15
Accessories	2,875.00	55
Furniture	12,000.00	8
Total	25,375.00	78

Sales by Month

VBA Macro

```
Sub GenerateReport()  
Dim ws As Worksheet  
Set ws = ThisWorkbook!  
Worksheets("Report")  
Data processing...  
ws.Range("A1").Value =  
"Sales Report"  
End Sub
```



K2 ERP

Search...

Home Sales Customers Documents Products Warehouse Reports Analytics Purchases Production Finance Projects Administration

Sales Document

General Items Additional Links History

Number: SO-000123

Date: 12.10.2026

Customer: Example LLC

Warehouse: Main warehouse

Status: In progress

Code	Product	Qty	Price	Amount
1001	Laptop	5	1,200.00	6,000.00
1002	Monitor	10	450.00	4,500.00
1003	Keyboard	2	85.00	170.00

+ Add item

Subtotal: 10,670.00

VAT: 2,134.00

Total: 12,804.00



Advantage: transition from file-based logic to a centralized data-driven system. All processes, data and analytics in one place.



Developed in Ukraine

A platform is stronger when it is built on real engineering experience.

erp.kyiv.ua

22

Excel → PostgreSQL + Python + K2

Sheets → entities and tables. Formulas and macros → analyzed rules in Python.
Reports → tools in K2 for data visualization.



1 Sheets → entities and tables

Data structure, relationships, types

ID	Product	Category	Quantity	Unit Price	Amount	Warehouse
1001	Laptop	Electronics	10	11,200.00	12,000.00	Kyiv
1002	Mouse	Accessories	5	450.00	2,250.00	Lviv
1003	Monitor	Electronics	8	8,500.00	17,000.00	Kharkiv
1004	Keyboard	Accessories	3	25,000.00	75,000.00	Kyiv



PostgreSQL
Entities and tables, relationships, constraints



2 Formulas and macros → analyzed rules in Python

Calculations, conditions, user logic

File Home Insert Page Layout Formulas Data Review View Developer Help

A1 =IF(E2="", "", SUMIFS(Sales!G:G, Sales!B:B, B2, Sales!E:E, E2))

1 Product Quantity Unit Price Amount VAT

2 Laptop 10 1,200.00 12,000.00 2,400.00

3 Mouse 5 450.00 2,250.00 450.00

9 Monitor 8 8,500.00 17,000.00 3,400.00

VBAProject (Sales.xlsm)

Microsoft Excel Objects

- Sheet1 (Sales)
- Sheet2 (Summary)
- Module1

Module1

```
Sub CalculateTotal()  
    Dim rng As Range  
    For Each rng In Range("E2:E100")  
        If rng.Value <> "" Then  
            rng.Offset(0,1).Value = rng.Value * 1.2  
        End If  
    Next rng  
End Sub
```



Python
Analyzed business rules, recreated logic, testing

```
def calculate_total(product_code, period):  
    df = db.session.query(func.sum(Sale.amount))\  
        .filter(Sale.product_code == product_code)\  
        .filter(Sale.date.between(period.start, period.end))  
    if df is None:  
        return 0  
    return df
```

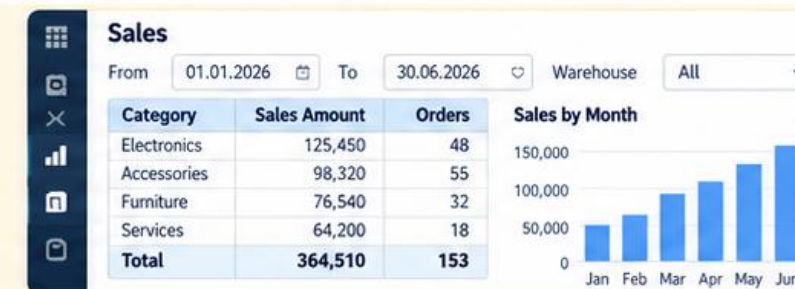
additional validations, error handling,
business rules and tests

3 Reports → K2 tools for data visualization

Combined tables, charts, analytics



K2
Reports, dashboards, analytics, export



Advantage: reuse existing work with centralized data. Formulas and VBA are reimplemented and tested, not just copied to a web interface.



Delphi: a form is not the main value



The core value of a Delphi solution is business logic, algorithms, data models and well-thought-out scenarios. They can be analyzed, preserved and reimplemented in a new architecture.

Example: a classic Delphi application (test sample)

1 Delphi form (VCL)

Code	Name	City
1	Example LLC	Kyiv
2	Alpha Ltd	Lviv
3	Beta LLC	Odesa

2 DFM fragment (text)

```
object frmCustomers: TfrmCustomers
  Left = 0
  Top = 0
  Caption = 'Customers (test example)'
  ClientHeight = 380
  ClientWidth = 480

  object edtName: TEdit
    Left = 88
    Top = 48
    Width = 273
    TabOrder = 0
  end

  object btnSave: TButton
    Left = 88
    Top = 216
    Width = 75
    Caption = 'Save'
    TabOrder = 2
   OnClick = btnSaveClick
  end
end
```

3 Pascal code fragment

```
procedure TfrmCustomers.btnSaveClick
(Sender: TObject);
begin
  if edtName.Text = '' then
  begin
    ShowMessage('Enter a name');
    Exit;
  end;

  qrCustomer.ParamByName('NAME')
    .AsString := edtName.Text;
  qrCustomer.ParamByName('CITY')
    .AsString := edtCity.Text;

  if edtID.Text = '' then
    qrCustomer.Insert
  else
    qrCustomer.Edit;

  qrCustomer.Post;
  ShowMessage('Data saved');
end;
```

New K2 architecture

Delphi (existing solution)

User interface
(VCL forms)

Business logic
algorithms,
functions, rules

Data models
structure,
relationships,
dictionaries

K2 (new architecture)

Modern web interface
TypeScript / Vue

API
REST (JSON)

Business logic
Python
services, algorithms,
business rules

Database
PostgreSQL
data model,
structure, relationships

Modern
web client
in browser

Users
employees,
partners, customers

Access
from any device



Advantage: preserving the business functionality when changing the client technology.



Delphi: how the logic is transferred



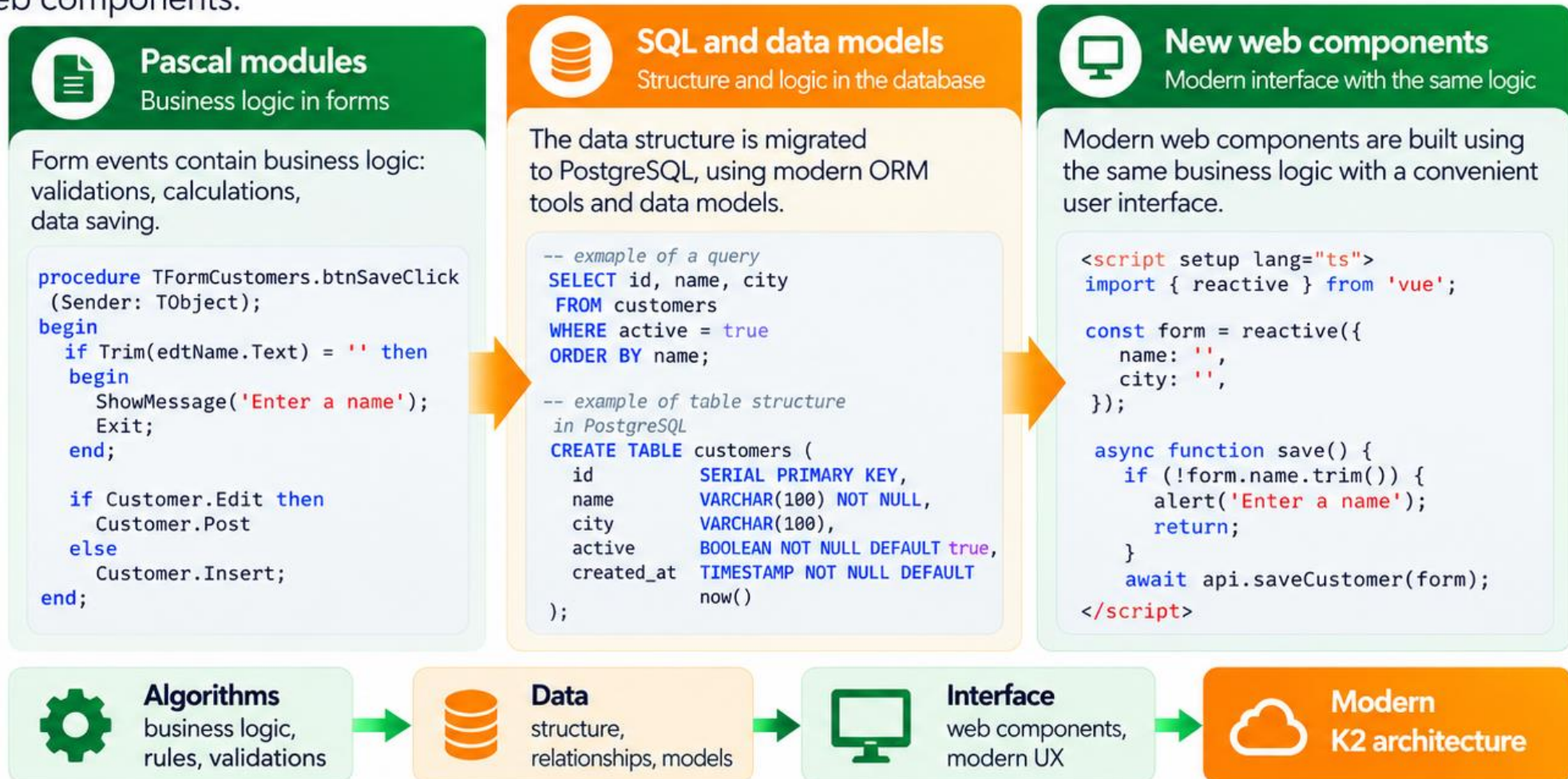
Pascal modules → analysis and transfer of algorithms.
SQL and data models → PostgreSQL and ORM.
VCL interface → new web components.

Logic migration is not just copying a form into another environment. It is important to discover the algorithms, data models and interface to reproduce the functionality in a modern architecture.



Advantage:

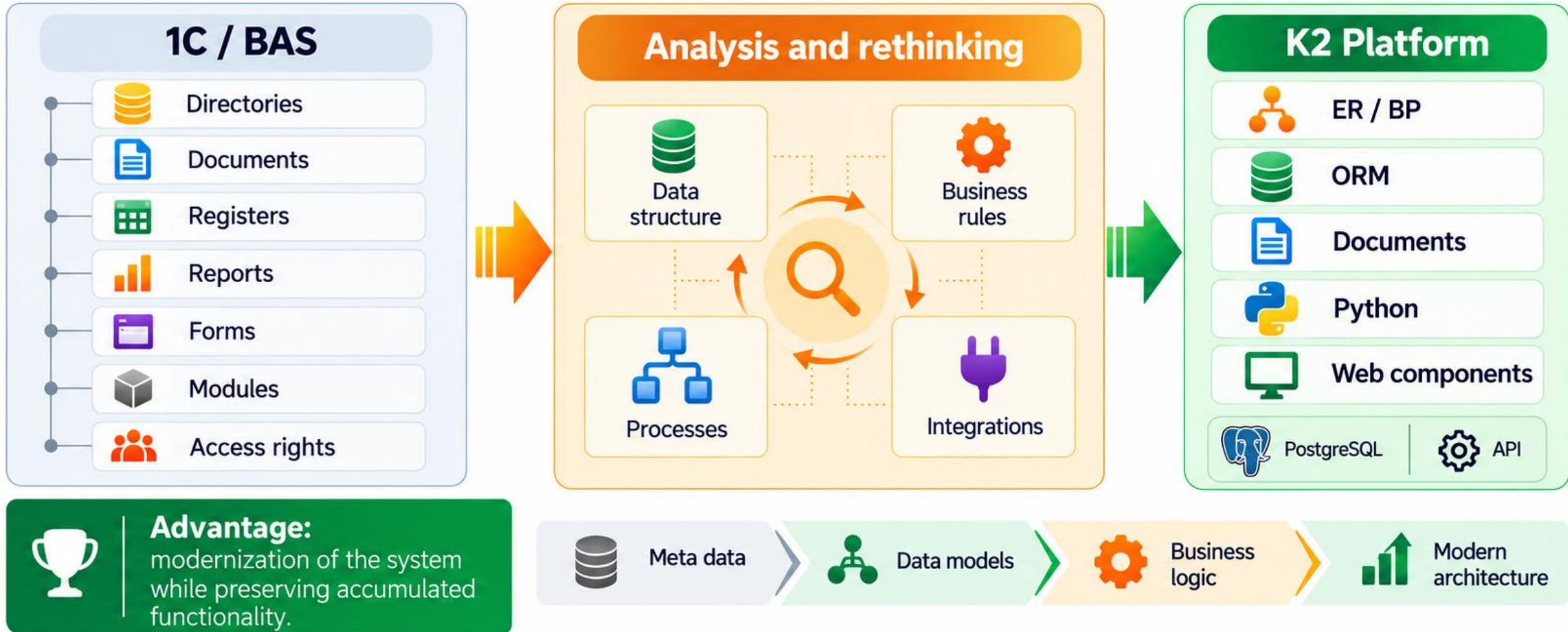
the ability to separate business logic from dated interface technology.



1C/BAS: preserve your data, upgrade the technology



Directories, documents, registers, reports and business rules are analyzed before migration.
The goal is to transfer the necessary functionality, not to copy every limitation of the legacy architecture.

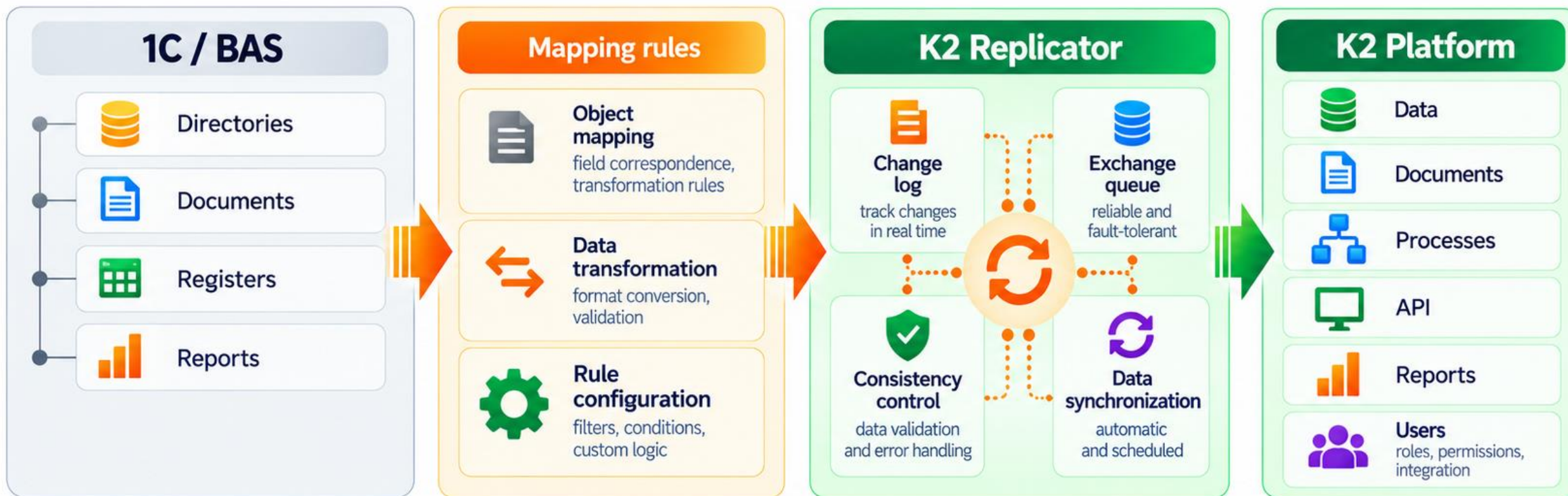


Replicator: migration without downtime



K2 Replicator supports data transfer and synchronization from 1C/BAS.

The new system can run in parallel with the legacy one with data exchange control and consistency.



Advantage:

the ability to perform a step-by-step migration without business interruption.



Parallel operation

run both systems simultaneously



Consistency control

data integrity and reconciliation



Gradual migration

step-by-step transition



Developed in Ukraine

erp.kyiv.ua

Migration without downtime is always easier than an emergency transition later.

Migration is not just a switch – it's parallel development



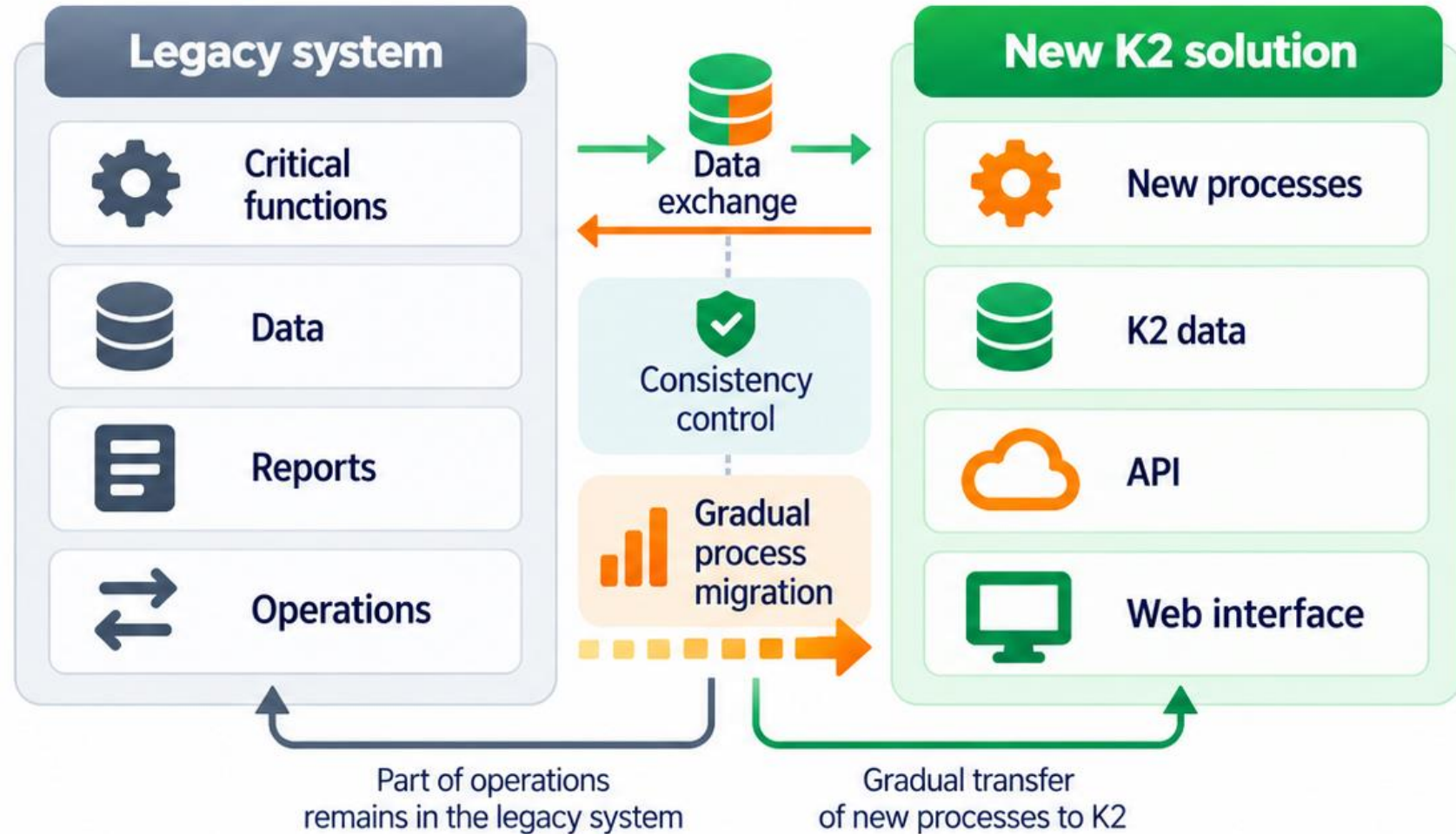
The legacy system **continues** to perform critical functions.

The new K2 solution receives **data** and gradually **takes** over the required processes.



Advantage:

reduced risks
of the migration project.



Business process renewal

Algorithms are often hidden in event handlers, macros and source code.

BPM models describe the processes for developers and end users, making them clear and understandable.



Legacy system: event handlers, macros, code

```
UnitDocument.pas x UnitReport.pas x Module1.bas x
1 procedure TfrbDocument.btnSaveClick(Sender: TObject);
2 begin
3     if edtNumber.Text = '' then
4         raise Exception.Create('Enter number');
5     if edtDate.Date = 0 then
6         raise Exception.Create('Enter date');
7
8     if not CheckBalance(edtProduct.Value, edtSum.Value) then
9         raise Exception.Create('Insufficient balance');
10
11     qry.Insert;
12     qry.FieldByName('NUM').AsString := edtNumber.Text;
13     qry.FieldByName('DATE').AsDateTime := edtDate.Date;
14     qry.FieldByName('SUM').AsFloat := edtSum.Value;
15     qry.FieldByName('CONTRAGENT_ID').AsInteger :=
16         edtCounterparty.KeyValue;
17     qry.Post;
18 end;
```

SQL queries in the legacy system

```
SELECT SUM(Quantity) AS Qty
FROM Stock
WHERE Product_ID = :Product_ID
AND Warehouse_ID = :WhID
AND Date <= :Date
GROUP BY Product_ID;
```

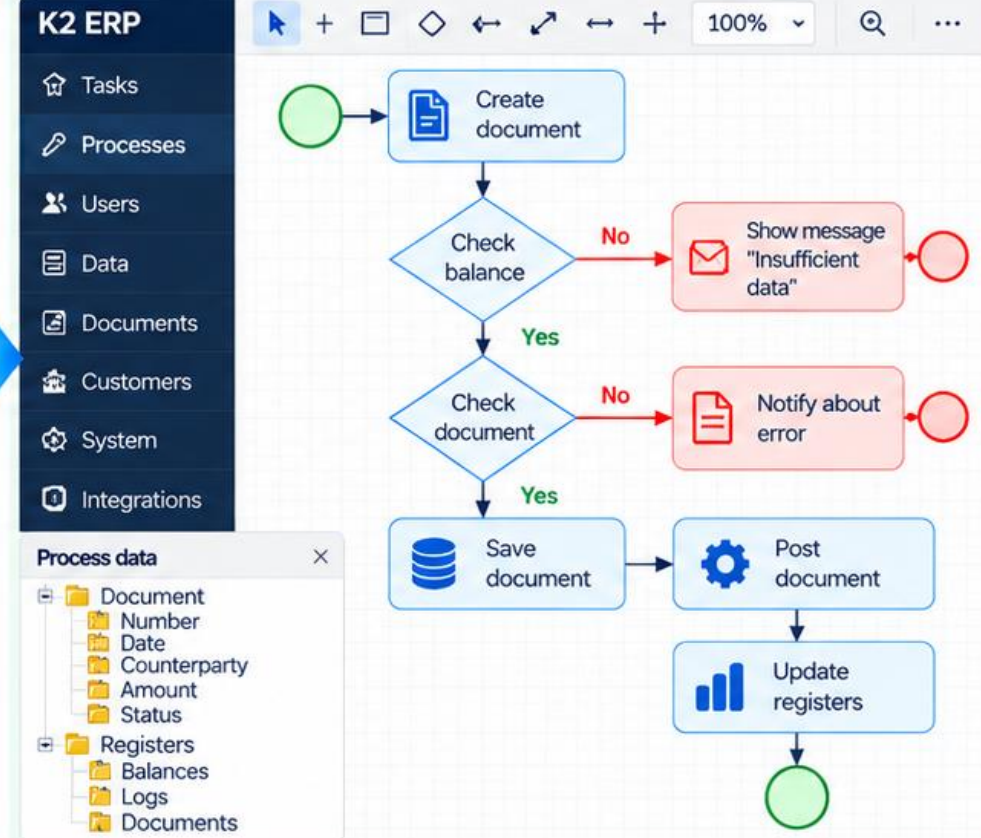
Macros / customization

- Documents
 - Event handlers
 - Calculations
 - Document validation
- Reports
 - Balance control
 - Document print
- Common modules
 - Calculations
 - Business rules
- Registers
 - Data processing
 - Form customization

Analysis and formalization of business rules

- Identification of key scenarios
- Analysis of algorithms and validations
- Data structure and relationships
- Business rules and conditions
- Identification of issues and alternative scenarios
- Alignment with developers and stakeholders

BPM process model in K2 ERP



Advantage: formalization and documentation of processes that are clear for developers and end users.



Developed in Ukraine

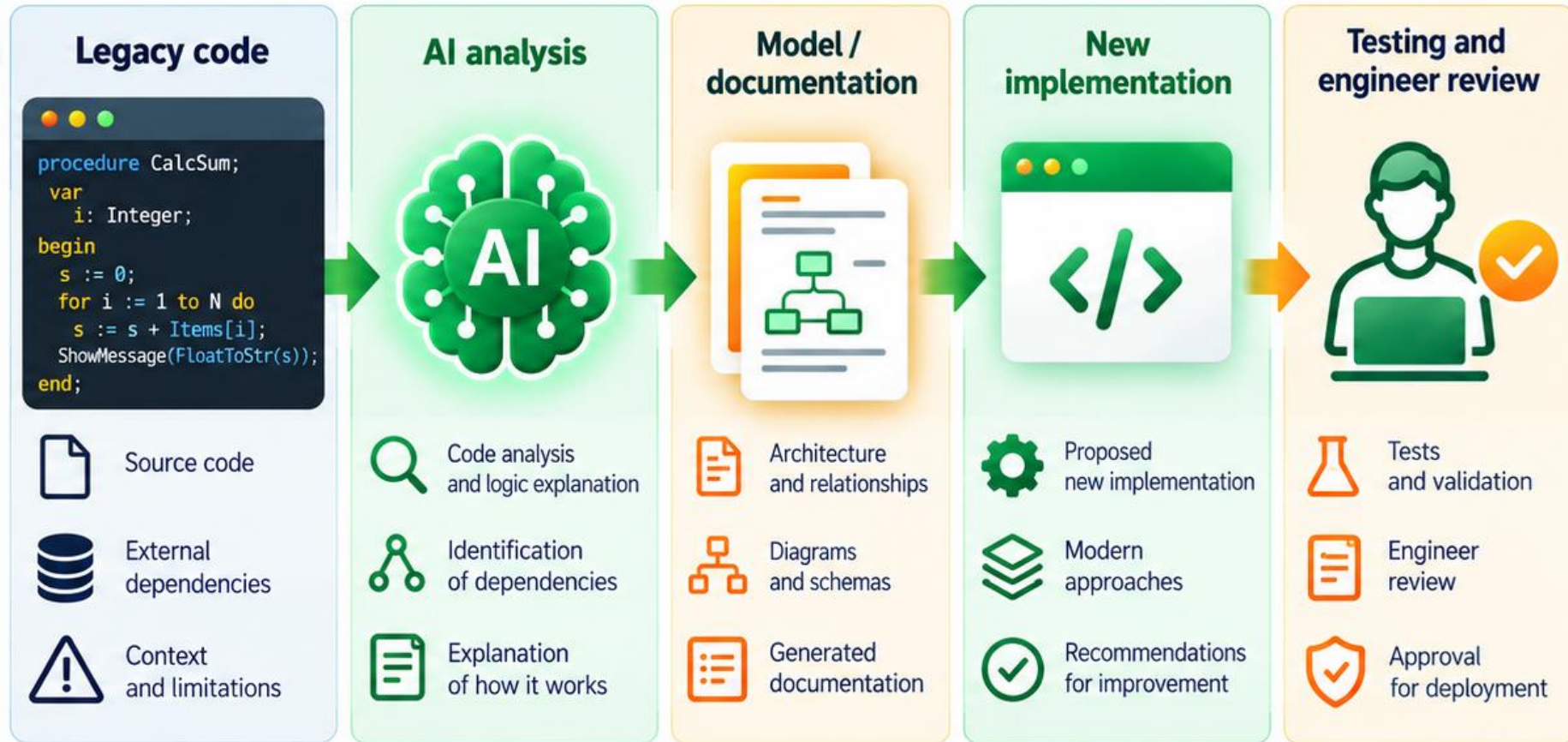
erp.kyiv.ua

A platform is stronger when it is built on real engineering experience.

AI helps analyse legacy systems



- **AI** can help analyse code, explain dependencies, generate documentation and propose a new implementation.
- The result always requires a review and validation by an engineer.



Advantage:

use of modern AI tools in reverse engineering.

AI helps, but the final decision is always with an engineer

Data verification after migration

The same input data. The same business rules. Control of results.
We compare calculations, documents, balances, access rights and critical scenarios.



What we verify



Documents

creation, posting, cancellation



Balances

items, materials, warehouses



Calculations

amounts, taxes, cost



User rights

roles, access, restrictions



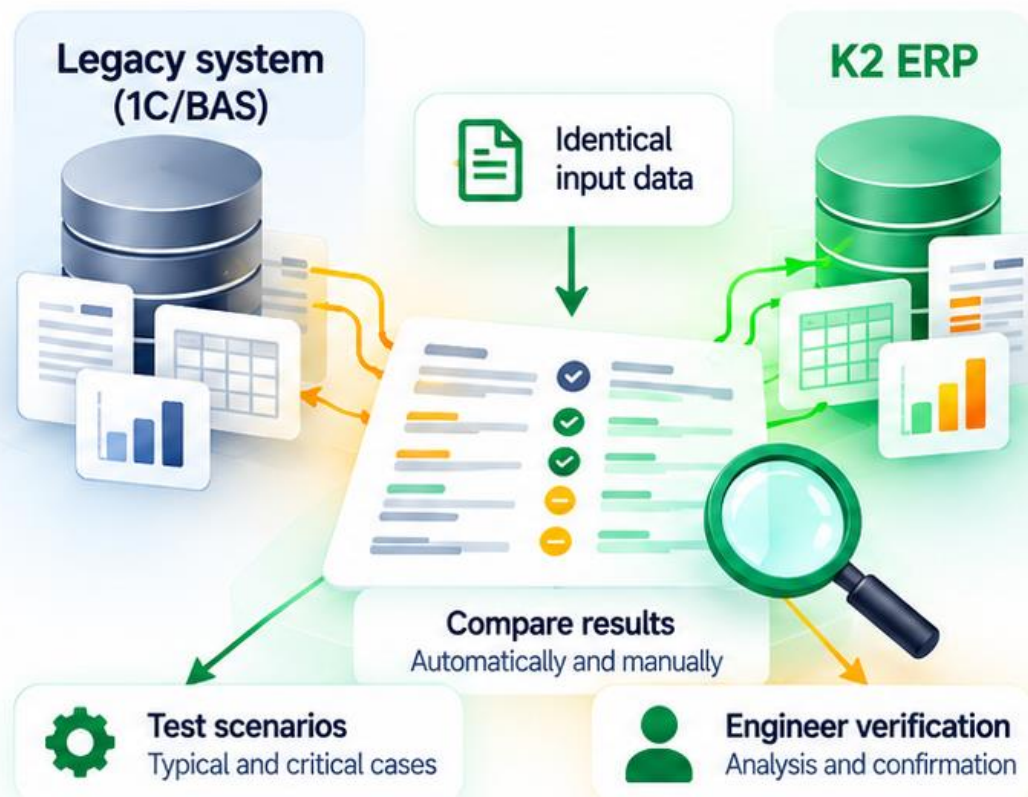
Reports

data formation and accuracy



Integrations

data exchange, external systems



Example of verification (conditional test data)

Object of verification	Legacy system (1C/BAS)	K2 ERP	Result
Sales document (total amount)	12 560,00	12 560,00	✓
Stock balance (item A)	100 / 25 / 0	100 / 25 / 0	✓
VAT calculation (Invoice)	10 466,67	10 466,67	✓
Document posting	Posted	Posted	✓
User rights	Access granted	Access granted	✓
Report generation (number of rows)	125 rows	125 rows	✓
Integration (data exchange)	10 / 10 / 0	10 / 10 / 0	✓



Advantage: reduces the risk of hidden functional errors after migration.



Modern web client instead of the old desktop client



Browser, different operating systems, modern components, reactive interface and the ability to work from mobile devices.

The old business logic is not required to stay in the old window.



Advantage:

separate user access from the old desktop environment.

Old desktop client



Local application



Tied to OS



Separate workplace

K2 web client



Access via browser from anywhere



The same business logic — more scenarios



Browser



Windows / Linux / macOS



Reactive interface



Mobile workplaces



Developed in Ukraine

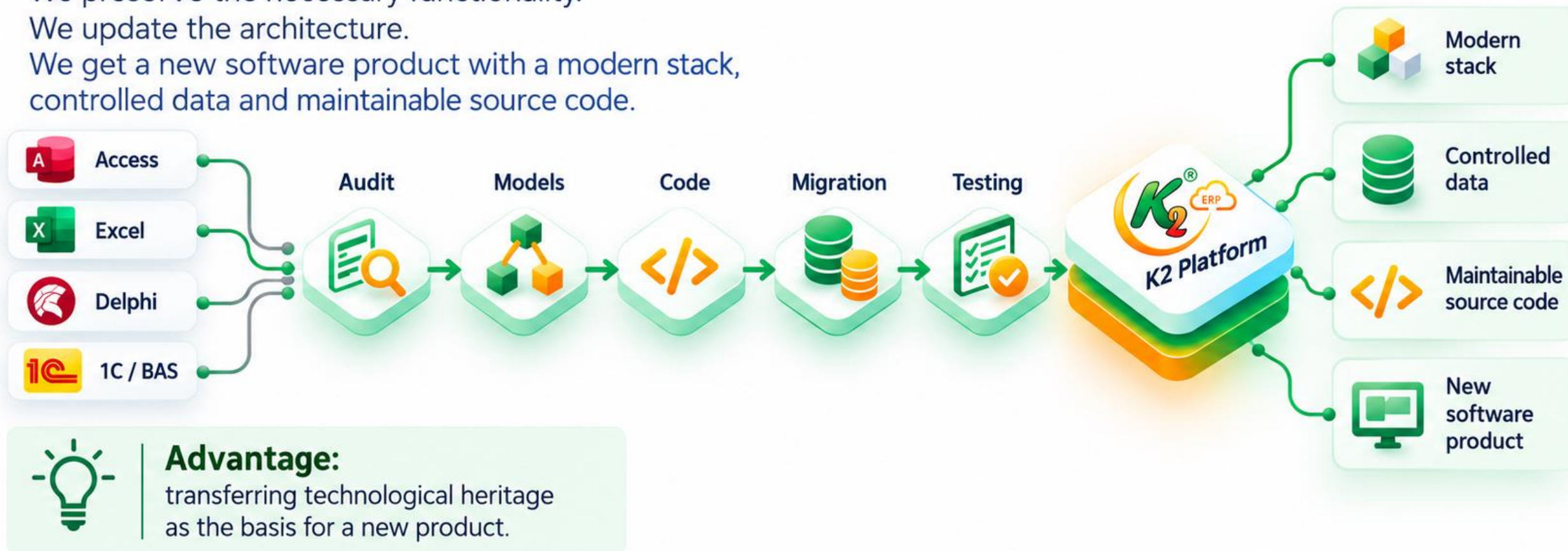
erp.kyiv.ua

A modern interface — more opportunities for your business.

Result of reverse engineering



We preserve the necessary functionality.
We update the architecture.
We get a new software product with a modern stack, controlled data and maintainable source code.



Visual editor of ER models

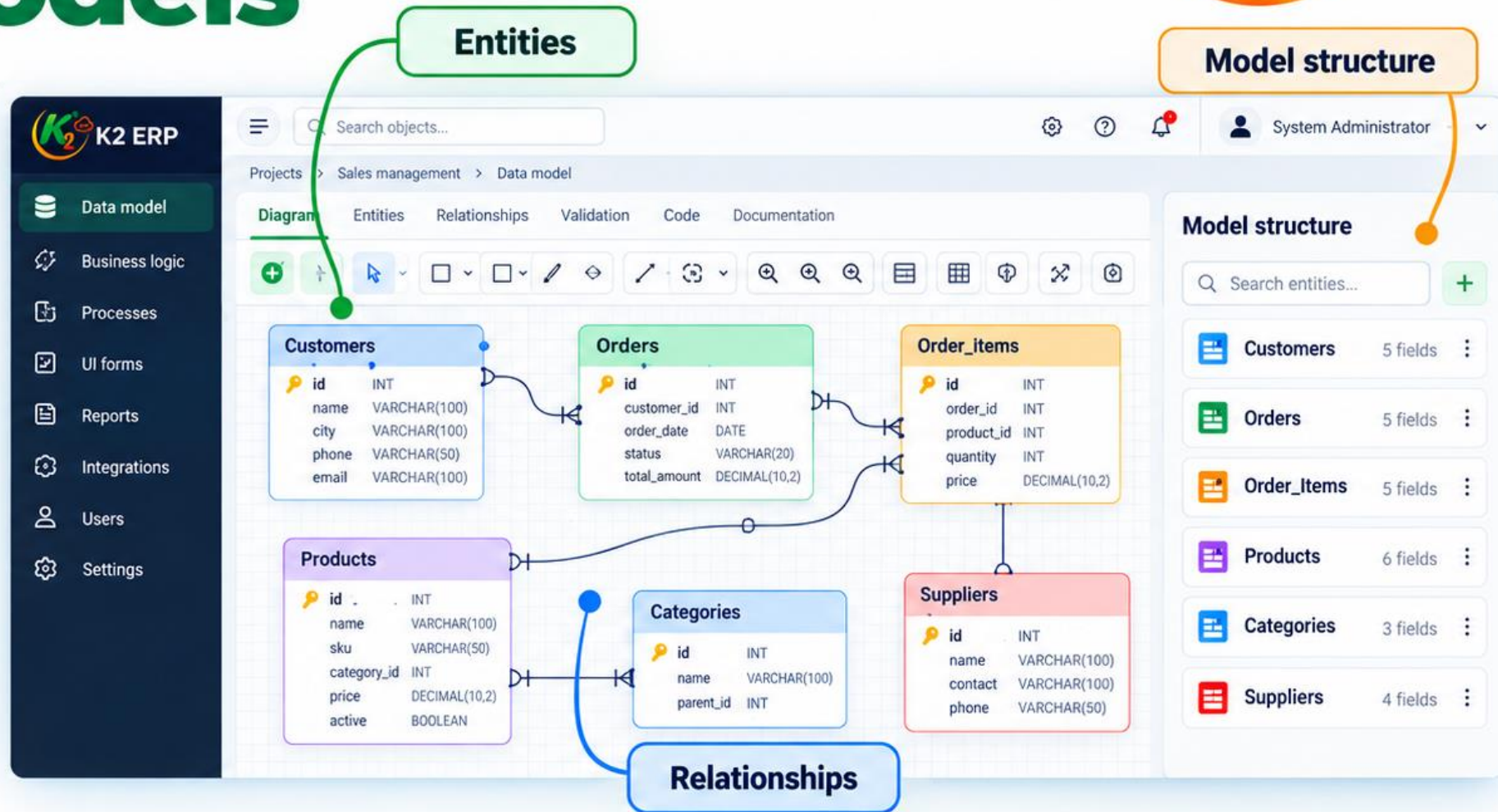


The K2 ER editor shows entities, their attributes and relationships in a graphical view.

The data model becomes a visible part of the application architecture.



Advantage:
analysis and design
of complex data
structures.

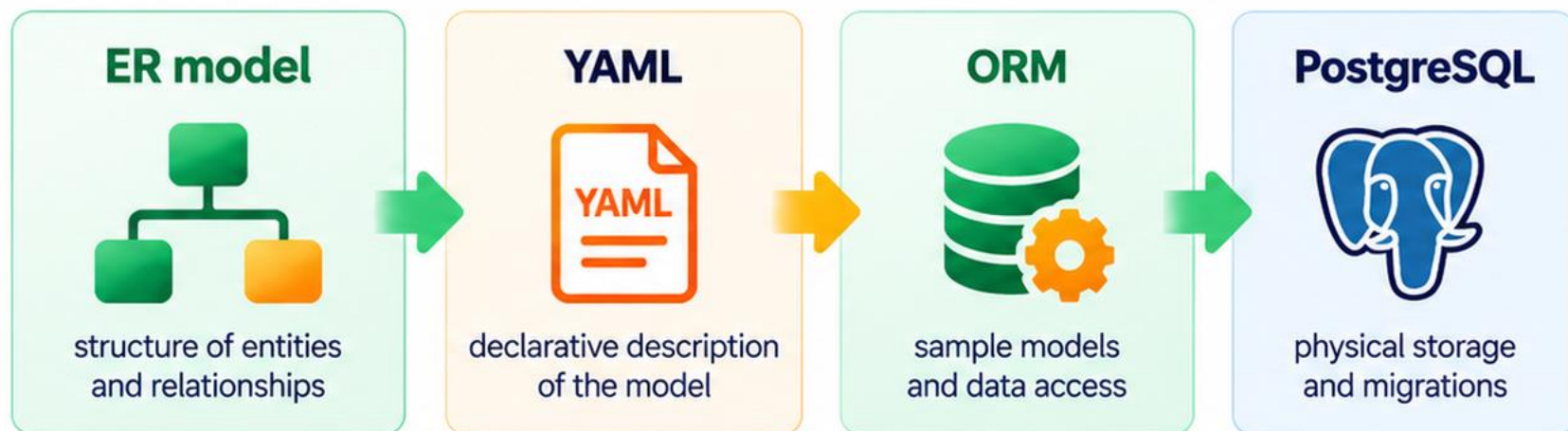


ER models: from schema to implementation



The data structure is described using YAML, ORM and migrations together.

A model is not just a nice picture. It is part of the software development process.



Advantage:

reduces the gap between design and implementation.

Fragment of an ER model



YAML description (example)

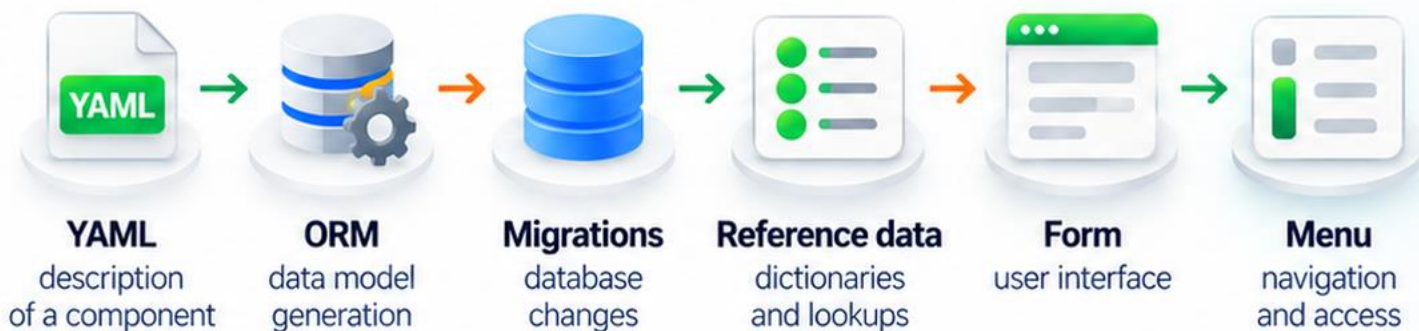
```
entities:
  orders:
    table: orders
    fields:
      id:
        type: integer
        primary: true
      customer_id:
        type: integer
        relation: customers.id
      order_date:
        type: date
      total_amount:
        type: decimal(10,2)
```

YAML — we describe a component declaratively



In YAML you can describe entities, fields, relationships, reference data, forms, menus and other component elements.

The description is human-readable and suitable for version control and generation.



```
entity: contractor
title: "Contractors"
type: directory
fields:
  id:
    type: integer
    primary_key: true
  code:
    type: string
    title: "Code"
    required: true
  name:
    type: string
    title: "Name"
    required: true
```

Entity

Component type

Fields



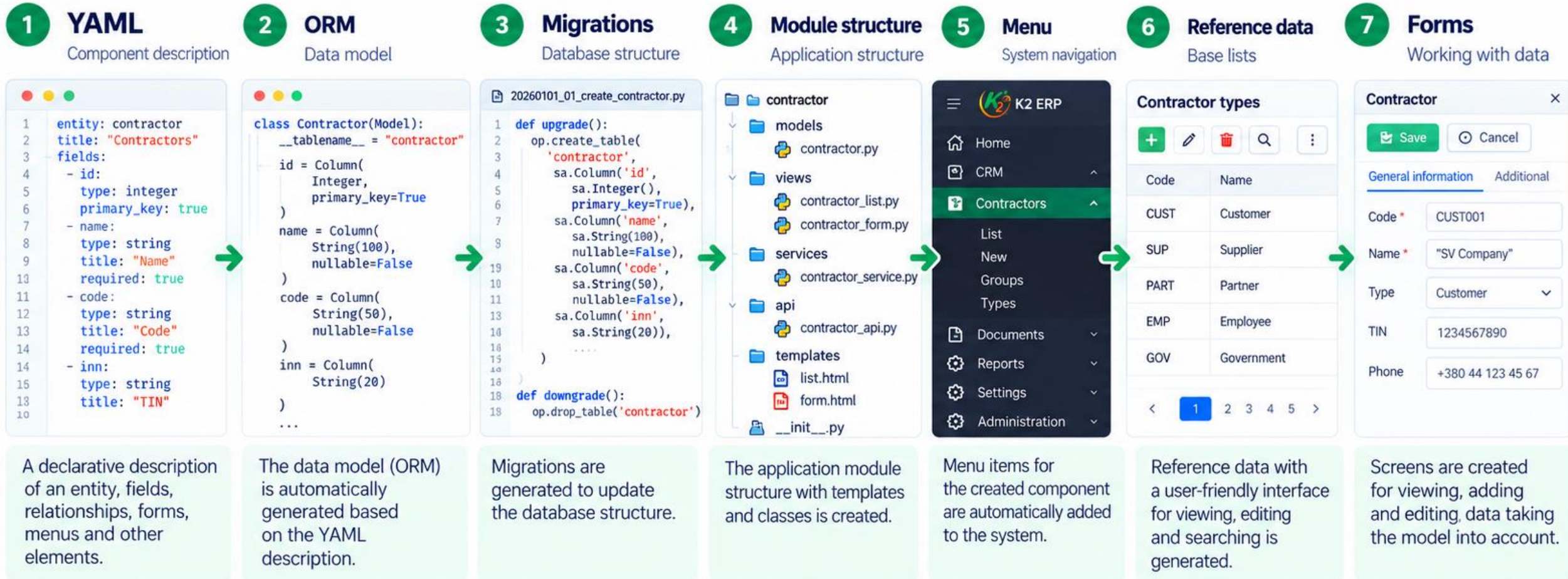
Advantage: structured description of components without the need for manual creation of each base part.

Application generation based on YAML



K2 uses a model to automate the typical stages of application creation:

YAML → ORM → migrations → module structure → menu → reference data → forms.



PostgreSQL: managed data storage

PostgreSQL is the main DBMS of K2.

Transactions, indexes, complex queries, large amounts of data and reliable storage mechanisms.



ACID
Reliable transactions



Indexes
High query performance



Complex queries
Analytics and reporting



Large volumes
Working with big datasets

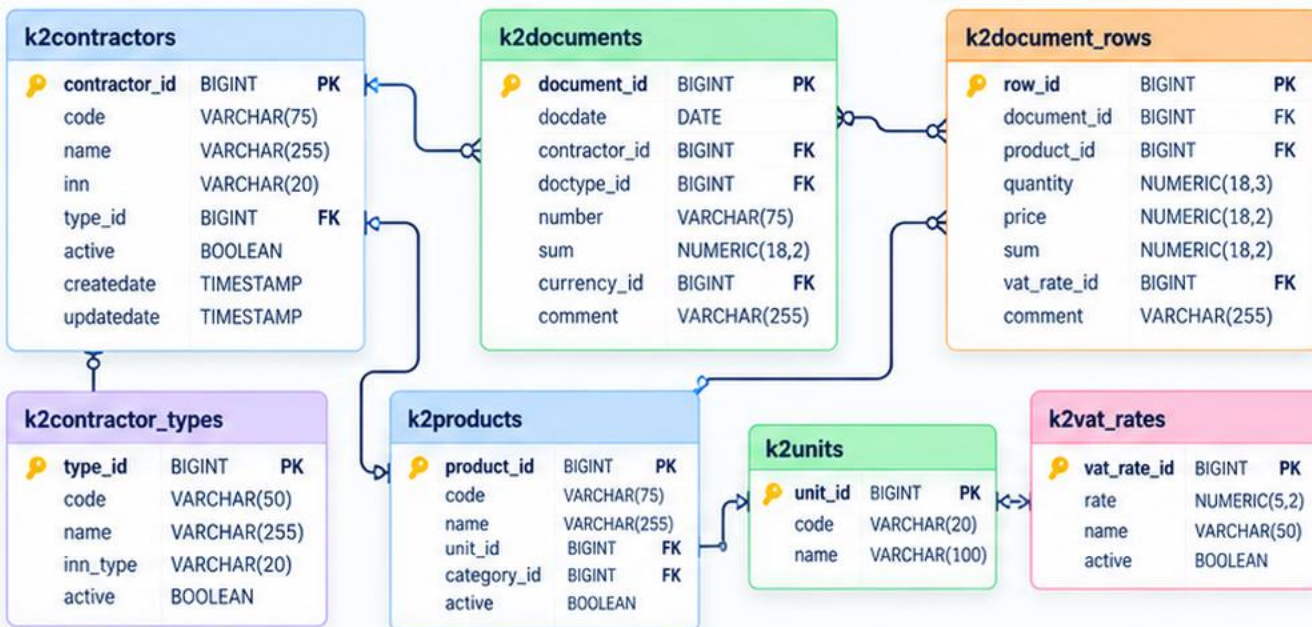


Reliability
Replication and backup



Scalability
Replication and clustering

Database structure in PostgreSQL (example)



Keys, indexes and relationships between tables are used to ensure data integrity and high performance.

Fragment of the real K2 database structure

- ▼ k2erp
 - ▼ Schemas (1)
 - ▼ public
 - ▼ Tables (8)
 - k2contractors
 - k2contractor_types
 - k2documents
 - k2document_rows
 - k2products
 - k2product_groups
 - k2units
 - k2vat_rates
 - Views (12)
 - Functions (56)
 - Sequences (8)
 - Types (34)
 - Indexes (45)
 - Foreign Tables (4)
 - Extensions (3)

Columns					Constraints	Indexes	Foreign Keys	Triggers
#	Name	Type	Nullable	Default				
1	contractor_id	bigint	not null	nextval('k2c...				
2	code	character varying(75)	not null					
3	name	character varying(255)	not null					
4	inn	character varying(20)	yes					
5	type_id	bigint	yes					
6	active	boolean	not null	true				
7	comment	character varying(255)	yes					
8	createdate	timestamp without time zone	not null	now()				
9	createuser	timestamp varying(75)	yes					
10	updatedate	timestamp without time zone	yes					
11	updateuser	character varying(75)	yes					

Indexes

Name	Definition	Unique
k2contractors_pkey	btree (contractor_id)	Yes
k2contractors_code_idx	btree (code)	No
k2contractors_inn_idx	btree (inn)	No



ORM — working with objects in Python



ORM connects K2 business objects with the database structure.
Developers work with Python models and use ready-made mechanisms for data access.

contractor.py

```
1 from k2.db import Model, Column, Integer, String, Boolean, DateTime, ForeignKey
2
3 class Contractor(Model):
4     __tablename__ = "k2contractors"
5
6     id = Column(Integer, primary_key=True)
7     code = Column(String(75), nullable=False, unique=True)
8     name = Column(String(255), nullable=False)
9     inn = Column(String(20))
10    type_id = Column(Integer, ForeignKey("k2contractor_types.type_id"))
11    active = Column(Boolean, default=True)
12    comment = Column(String(255))
13    createdate = Column(DateTime)
14    createuser = Column(String(75))
15    updatedate = Column(DateTime)
16    updateuser = Column(String(75))
17
18    type = relationship("ContractorType", back_populates="contractors")
19
20 class ContractorType(Model):
21     __tablename__ = "k2contractor_types"
22
23     type_id = Column(Integer, primary_key=True)
24     code = Column(String(50), nullable=False, unique=True)
25     name = Column(String(255), nullable=False)
26     inn_type = Column(String(20))
27     active = Column(Boolean, default=True)
28     contractors = relationship("Contractor", back_populates="type")
```

Python model
of an entity

ORM

Maps the model
to database tables
and relationships

Working with data

Creation, reading,
updating, deletion
through ORM

Relationship
with another entity

Tables in PostgreSQL



k2contractors

Column	Type	NULL	Key	Description
contractor_id	bigint	no	PK	Identifier
code	varchar(75)	no	UQ	Code
name	varchar(255)	no		Name
inn	varchar(20)	yes		Tax ID (INN)
type_id	bigint	no	FK	Contractor type
active	boolean	no		Active
comment	varchar(255)	yes		Comment
createdate	timestamp	yes		Creation date
createuser	varchar(75)	yes		Created by
updatedate	timestamp	yes		Update date
updateuser	varchar(75)	yes		Updated by

k2contractor_types

Column	Type	NULL	Key	Description
type_id	bigint	no	PK	Identifier
code	varchar(50)	no	UQ	Code
name	varchar(255)	no		Name
inn_type	varchar(20)	yes		Tax ID type
active	boolean	no		Active



Database migrations



The database structure changes together with the component development.

ORM and migrations help control schema changes between versions.



Advantage:

reproducible and controlled schema changes.



Compare changes



Validate migrations



Apply changes



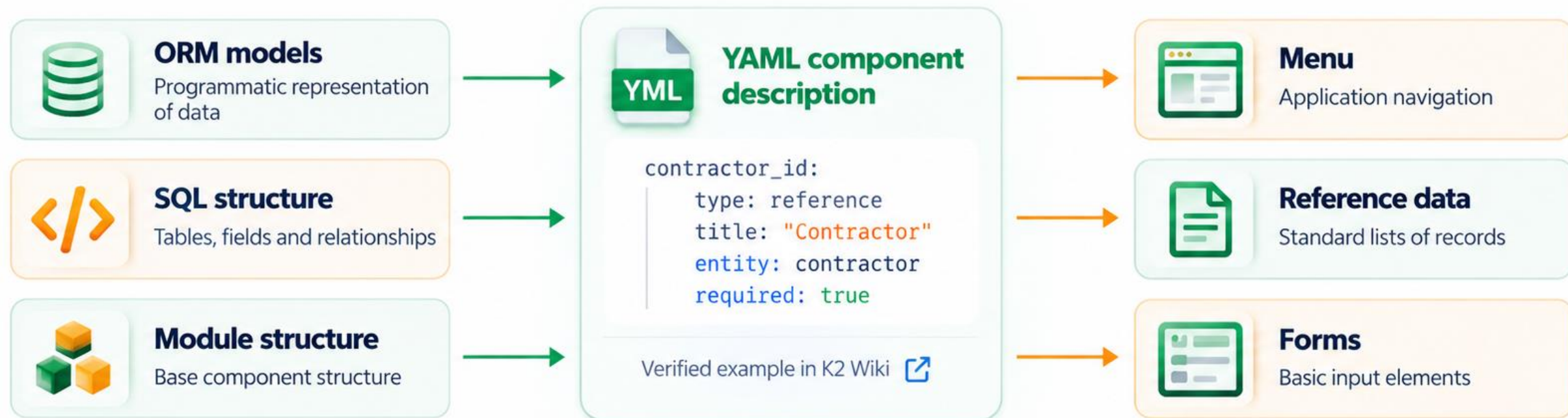
New database schema

One model — multiple software artifacts



From a component description you can generate database structures, program models, menus, reference data, forms and basic operations.

The developer focuses on business logic.



Advantage: reusable formalized model that generates multiple artifacts.

Reference data: no need to create it again



Lists, cards, fields, search, standard actions and relationships with other objects.

K2 constructors and components help you create reference data quickly.

Home

CRM

Contractors

List

New

Groups

Types

Documents

Reports

Reference data

Settings

Search

Quick search in the list by code, name or other fields.

Standard actions

Create, edit, delete, update, filters.

Code	Name	Type	Tax ID	City	Phone	E-mail
CUST001	Example LLC	Customer	1234567890	Kyiv	+380 44 123 45 67	info@example.ua
CUST002	Postach LLC	Supplier	9876543210	Dnipro	+380 56 234 56 78	postach@example.ua
CUST003	Service PE	Partner	1122334455	Lviv	+380 32 345 67 89	service@example.ua
CUST004	Trade LLC	Customer	9988776655	Odesa	+380 48 456 78 90	trade@example.ua
CUST005	State Org	Government	0000000000	Kyiv	+380 44 987 65 43	gov@example.ua
CUST006	Logistics LLC	Customer	6677889900	Kharkiv	+380 57 567 89 01	logistics@example.ua
CUST007	Producer LLC	Supplier	4455667788	Zaporizhzhia	+380 61 678 90 12	prod@example.ua
CUST008	Retail LLC	Customer	3344556677	Vinnysia	+380 43 789 01 23	retail@example.ua
CUST009	BuildService	Partner	5566778899	Poltava	+380 53 890 12 34	build@example.ua
CUST010	Innovation LLC	Customer	2233445566	Cherkasy	+380 47 901 23 45	innov@example.ua

1

2

3

4

5

...

>

1 - 10 of 356

10

Contractor: Example LLC

Save

Cancel

...

General information

Additional

Contacts

Accounts

Relationships

History

Code *

CUST001

Name *

Example LLC

Type *

Customer

Tax ID

1234567890

Company ID

123456789

Phone

+380 44 123 45 67

E-mail

info@example.ua

City

Kyiv

Group

Clients

Active

Contacts

2

Ivanenko Ivan

Director

+380 44 123 45 67

Petrenko Olena

Manager

+380 44 765 43 21

Contracts

3

Contract No. 1 from 01.01.2026

Contract No. 2 from 15.03.2026

Contract No. 3 from 10.06.2026

Invoices

5

UA12345678900001

>

UA12345678900002

>

UA12345678900003

>



Advantage: automation of routine CRUD development.



Relationships with other objects

Contacts, contracts, invoices and other related data.



Developed in Ukraine

Ready-to-use components for business applications.

erp.kyiv.ua

47

Document journal

Documents and document journals

A document is more than just a record in a table.

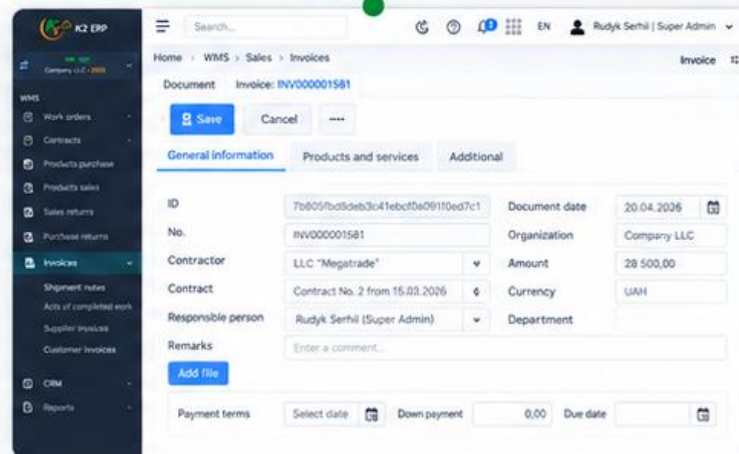
K2 provides mechanisms for documents, their journals, forms, relationships and business logic processing.



Advantage:

ready-to-use building block for document-oriented systems.

Document form



Document: Invoice: INV000001581

Buttons: Save, Cancel, ...

Tabs: General information, Products and services, Additional

Fields:

- ID: 7b805fbd3deb3c41ebc0e09110ed7c1
- No.: INV000001581
- Contractor: LLC "Megatrade"
- Contract: Contract No. 2 from 15.03.2026
- Responsible person: Rudyk Serhil (Super Admin)
- Remarks: (Enter a comment...)
- Payment terms: Select date
- Down payment: 0,00
- Due date: (calendar icon)

Document date: 20.04.2026

Organization: Company LLC

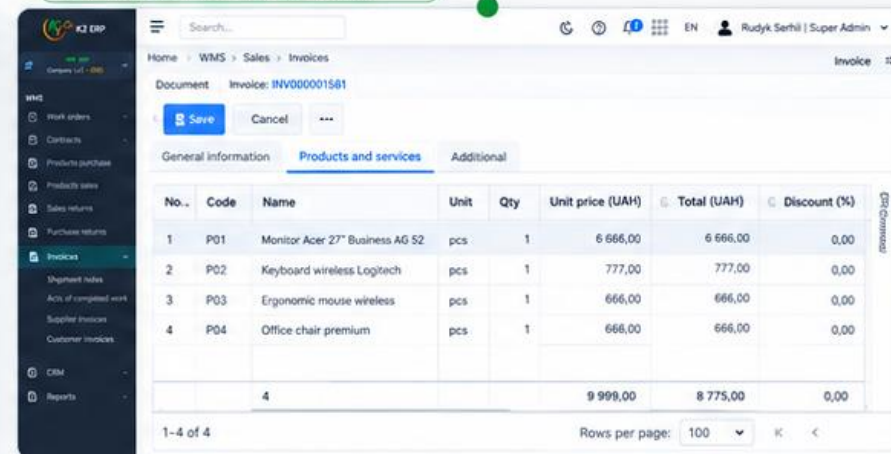
Amount: 28 500,00

Currency: UAH

Department: (empty)

Document lines

Products / related records



Document: Invoice: INV000001581

Buttons: Save, Cancel, ...

Tabs: General information, Products and services, Additional

No.	Code	Name	Unit	Qty	Unit price (UAH)	Total (UAH)	Discount (%)
1	P01	Monitor Acer 27" Business AG 52	pcs	1	6 666,00	6 666,00	0,00
2	P02	Keyboard wireless Logitech	pcs	1	777,00	777,00	0,00
3	P03	Ergonomic mouse wireless	pcs	1	666,00	666,00	0,00
4	P04	Office chair premium	pcs	1	666,00	666,00	0,00
					9 999,00	8 775,00	0,00

1-4 of 4

Rows per page: 100

Additional characteristics of objects



You can add additional characteristics to contractors, products, equipment or other entities. This allows you to adapt the data model to the specific needs of each client.



Advantage:

customization of domain data without developing a separate module for each characteristic.

The screenshot displays the K2 ERP interface for the 'Contractor' entity. The left sidebar shows the navigation menu with options like Home, CRM, Contractors, Products, Sales, Purchase, Warehouse, Documents, Reports, Reference data, and Settings. The main area shows the 'Contractor' entity card with tabs for 'Entity card', 'Related documents', and 'Procedures'. The 'Entity card' tab is active, showing a form with 'Base fields' (ID, Document No., Contractor, Responsible person, Contract, Parent organization, Notes) and 'Additional characteristics' (Payment terms, Advance amount, Priority, Delivery time (days), Discount (%)). A green box highlights the 'Base fields' section, and another green box highlights the 'Additional characteristics' section. A green arrow points from the 'Additional characteristics' section to the '+ Add characteristic' button.

Home > CRM > Contractors > Contractor

Contractor

Entity card | Related documents | Procedures

Save | Cancel | [Icons]

Base fields

ID	7b8b45fe8d3e1bc4se04e9f8d6e91c0e7c11	Document date	20.04.2026
Document No.	PR00001541	Organization *	K2 LLC
Contractor *	LLC "Megatrade"	Warehouse	Main warehouse
Responsible person	Enter a value or search...	Settlement account	PrivatBank (UA733052990000260812611704)
Contract	Enter a value or search...	Currency	UAH
Parent organization	Enter a value or search...		
Notes			

Add file

Additional characteristics

Payment terms	Select date	Delivery time (days)	0
Advance amount		Discount (%)	0
Priority			

+ Add characteristic

Attachments and files related to data



Contracts, photos, technical instructions, certificates and other files can be linked to business objects. Files become part of the information context.

K2 ERP

K2 LLC - 2026
Main office

WMS

- Tasks
- Contracts
- Products purchase
- Products sales
- Contractors**
 - List
 - New
 - Groups
 - Types
- Documents
- Reports
- Reference data
- Settings

Home > Contractors > Contractor: Example LLC

Contractor: Example LLC

Save Cancel

General information

AdditionalContactsAccountsInvoicesRelationshipsHistory

Code *CUST002

Name *Example LLC

Type *Supplier

Tax ID9876543210

Company ID987654321

Phone+380 56 234 56 78

E-mailpostach@example.ua

CityDnipro

GroupSuppliers

Active

Files

Add file

Adding files

Upload contracts, instructions, photos, certificates and other documents.

Related files

Files are stored together with the object and are directly accessible from the card.

	Name	Size	Type	Date
	Contract Nô.12-2026.pdf	1,2 MB	application/pdf	12.04.2026
	Quality certificate.pdf	856 KB	application/pdf	12.04.2026
	User manual.docx	340 KB	application/vnd.openxmlformats-...	10.04.2026
	Equipment photo.jpg	2,1 MB	image/jpeg	07.04.2026



Advantage: reusable unified attachment mechanism.



Developed in Ukraine

Files become part of the business object context.

erp.kyiv.ua

50

Python

as the language of K2 business logic

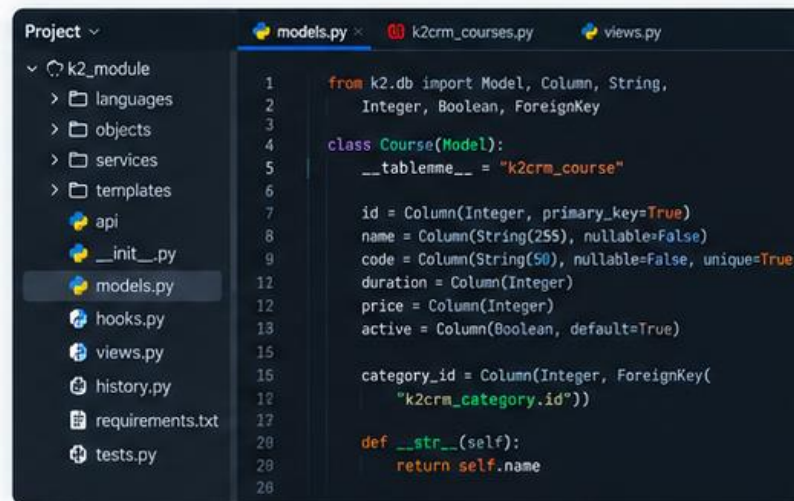
Business rules, data processing, calculations, handlers and integration logic can be implemented in Python.

There is no need to learn a proprietary language.



Advantage:

use of a widely adopted language to extend corporate systems.



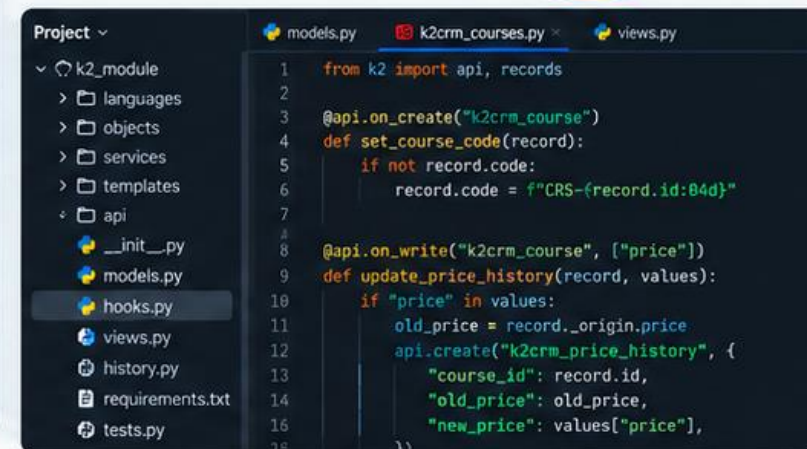
```

1 from k2.db import Model, Column, String,
2   Integer, Boolean, ForeignKey
3
4 class Course(Model):
5     __tablename__ = "k2crm_course"
6
7     id = Column(Integer, primary_key=True)
8     name = Column(String(255), nullable=False)
9     code = Column(String(50), nullable=False, unique=True)
10    duration = Column(Integer)
11    price = Column(Integer)
12    active = Column(Boolean, default=True)
13
14    category_id = Column(Integer, ForeignKey(
15        "k2crm_category.id"))
16
17    def __str__(self):
18        return self.name
  
```



Python models

Creating entities, relationships, constraints, business logic.



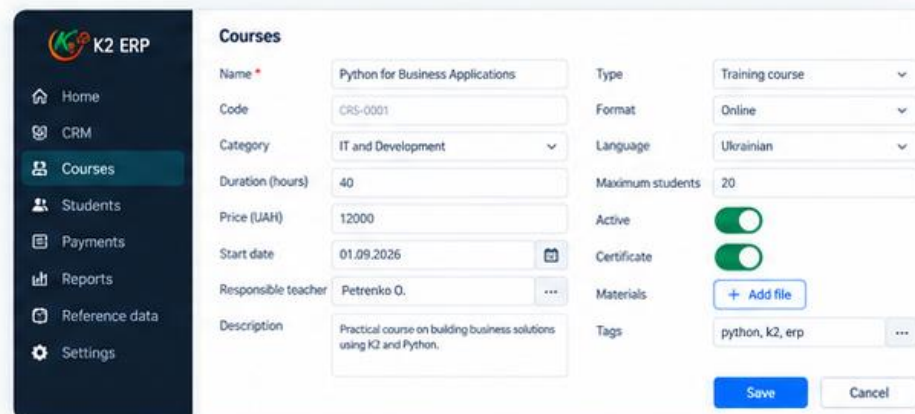
```

1 from k2 import api, records
2
3 @api.on_create("k2crm_course")
4 def set_course_code(record):
5     if not record.code:
6         record.code = f"CRS-{record.id:04d}"
7
8 @api.on_write("k2crm_course", ["price"])
9 def update_price_history(record, values):
10    if "price" in values:
11        old_price = record.origin.price
12        api.create("k2crm_price_history", {
13            "course_id": record.id,
14            "old_price": old_price,
15            "new_price": values["price"],
16        })
  
```



Routes and handlers

UI representation, controllers, request processing.



K2 ERP

Courses

Name: Python for Business Applications | Type: Training course

Code: CRS-0001 | Format: Online

Category: IT and Development | Language: Ukrainian

Duration (hours): 40 | Maximum students: 20

Price (UAH): 12000 | Active: ☒

Start date: 01.09.2026 | Certificate: ☒

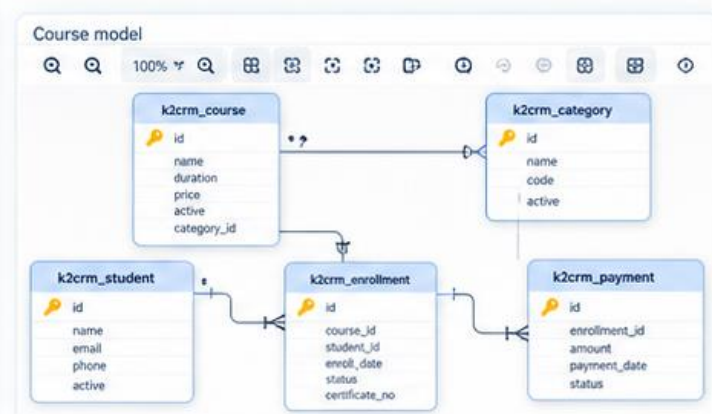
Responsible teacher: Petrenko O. | Materials:

Description: Practical course on building business solutions using K2 and Python. | Tags: python, k2, erp



Real component implementation

Forms, data validation and processing using Python logic.



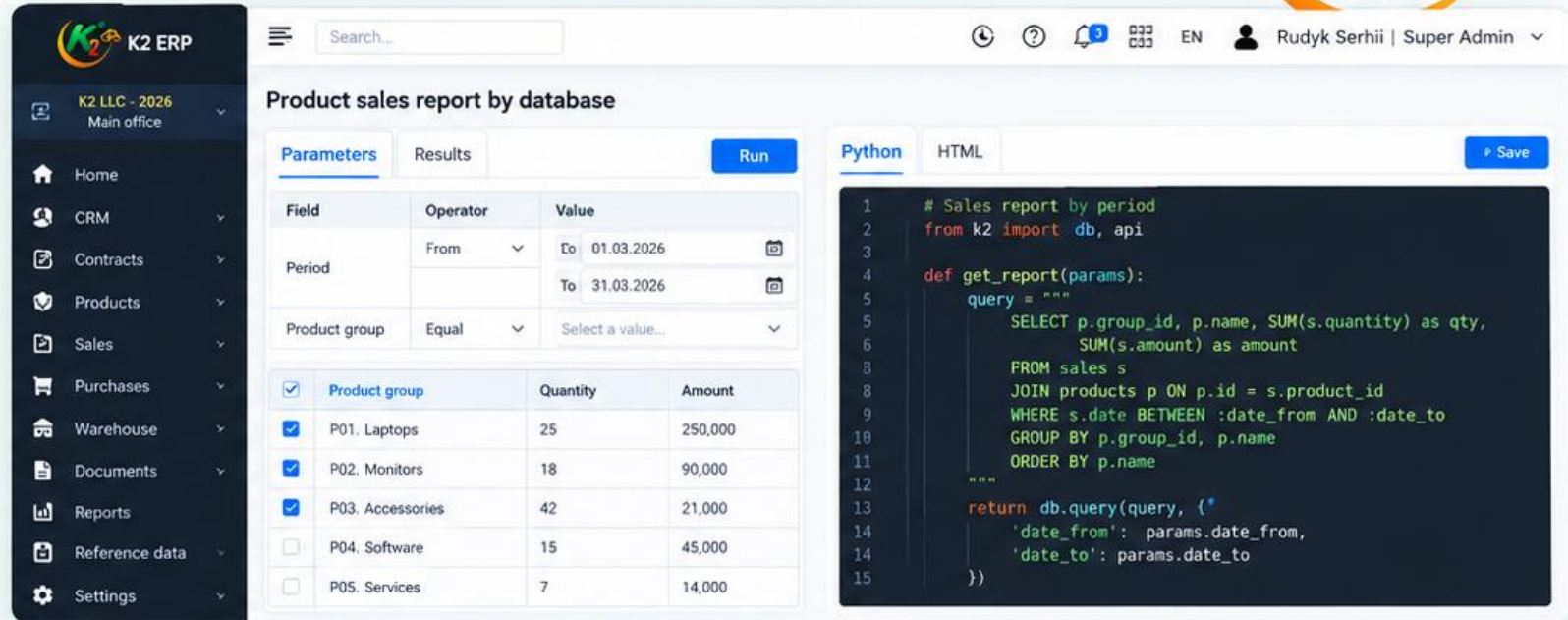
Component models and processes

Data schema represented in a visual model with Python logic.

Sample code and ready infrastructure

K2 provides a foundation for working with data and components. Python code complements the platform where no specific business logic is required.

Example of Python code and a ready-to-use component



Product sales report by database

Parameters: Period (From: 01.03.2026, To: 31.03.2026), Product group (Equal)

Field	Operator	Value
Period	From	To: 01.03.2026
	To	To: 31.03.2026
Product group	Equal	Select a value...

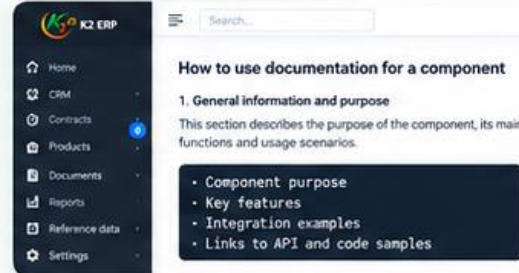
Results:

Product group	Quantity	Amount
☒ P01. Laptops	25	250,000
☒ P02. Monitors	18	90,000
☒ P03. Accessories	42	21,000
☐ P04. Software	15	45,000
☐ P05. Services	7	14,000

Python code:

```
1 # Sales report by period
2 from k2 import db, api
3
4 def get_report(params):
5     query = """
6         SELECT p.group_id, p.name, SUM(s.quantity) as qty,
7               SUM(s.amount) as amount
8         FROM sales s
9         JOIN products p ON p.id = s.product_id
10        WHERE s.date BETWEEN :date_from AND :date_to
11        GROUP BY p.group_id, p.name
12        ORDER BY p.name
13    """
14    return db.query(query, {
15        'date_from': params.date_from,
16        'date_to': params.date_to
17    })
```

Documentation + API



How to use documentation for a component

1. General information and purpose
This section describes the purpose of the component, its main functions and usage scenarios.

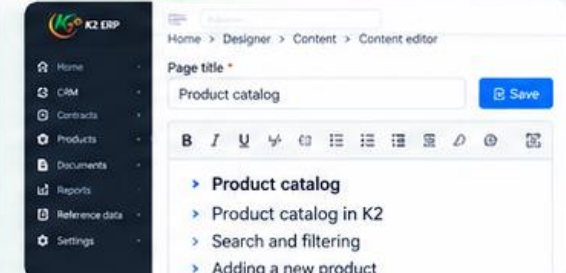
- Component purpose
- Key features
- Integration examples
- Links to API and code samples

Diagrams



Home > Designer > Diagrams > Business process

Content editor



Home > Designer > Content > Content editor

Page title: Product catalog

- > Product catalog
- > Product catalog in K2
- > Search and filtering
- > Adding a new product



Advantage:

focus on business tasks instead of infrastructure development.



When the infrastructure is already ready, you can focus on what really matters in the code.



Ready-to-use K2 mechanisms
Data, components, integrations, API



Custom Python logic
Business-specific rules and logic

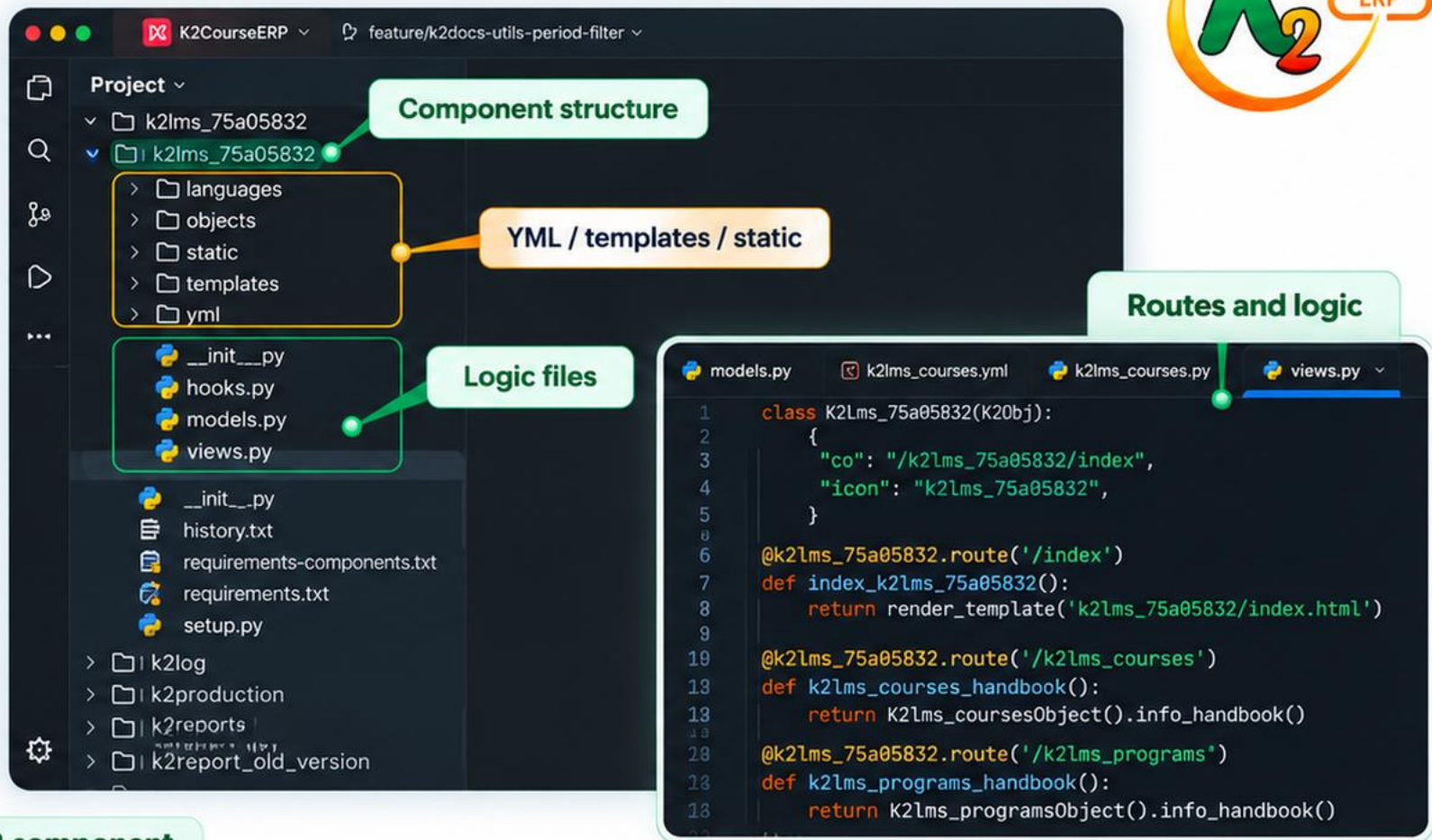


Business solutions
Automation, analytics, efficiency

Component structure of a product

K2 allows you to create your own modules, components, forms, reports and integrations.

Components can be developed and reused in different solutions.



Advantage:

modularity and reuse of development efforts.

K2 component



Component catalog in the real K2



The component structure gives developers a clear understanding of the composition and organization of the software system.

Knowing the structure makes it easier to maintain and extend solutions.



Advantage:

Clear navigation by the structure of software components.



Reactive web interface

TypeScript and Vue allow you to create interactive components whose interface reacts to state changes.

Less unnecessary page reloads — a more natural and smoother user experience.



Advantage:
modern reactive
client architecture.

The screenshot displays the K2 ERP web interface. The top navigation bar includes a search bar, user profile (Rudyk Serhii | Super Admin), and language (EN). The left sidebar shows the main menu with categories like WMS, Tasks, Contracts, Products, Sales, and Invoices. The main content area shows the 'Invoice' form for document PB000001541. The form includes fields for ID, Document No., Counterparty, Counterparty details, Contract, Document date, Organization, Warehouse, Organization details, and Currency. A green callout box labeled 'Live form state' points to the form fields. Below the form, a table shows the 'Document lines' with columns: No., Code, Item, Specification, Unit, Quantity, Price (excl. VAT), Amount (excl. VAT), Amount (VAT), and Total amount. The table contains 4 rows of data, including items like 'Monoblock Artline Business AIO S2', 'Keyboard KX(C)lip', 'Microcomputer MeGo pad', and 'External optical drive Asus SBW-0602X-U'. A green callout box labeled 'Updated content' points to the table. An orange callout box labeled 'No full page reload' points to the bottom of the page, indicating that the content is updated without a full page reload. The bottom of the page shows pagination (1 to 4 of 4) and page size (100).

No.	Code	Item	Specification	Unit	Quantity	Price (excl. VAT)	Amount (excl. VAT)	Amount (VAT)	Total amount
1	0/6	Monoblock Artline Business AIO S2		pcs	6	6,666.00	39,996.00	0.00	39,996.00
2	0/1	Keyboard KX(C)lip		pcs	1	6,666.00	6,666.00	0.00	6,666.00
3	0/1	Microcomputer MeGo pad		pcs	1	77.00	77.00	0.00	77.00
4	0/1	External optical drive Asus SBW-0602X-U		pcs	1	666.00	666.00	0.00	666.00
					9		39 996,00	0,00	39 996,00

Adaptive to screen sizes

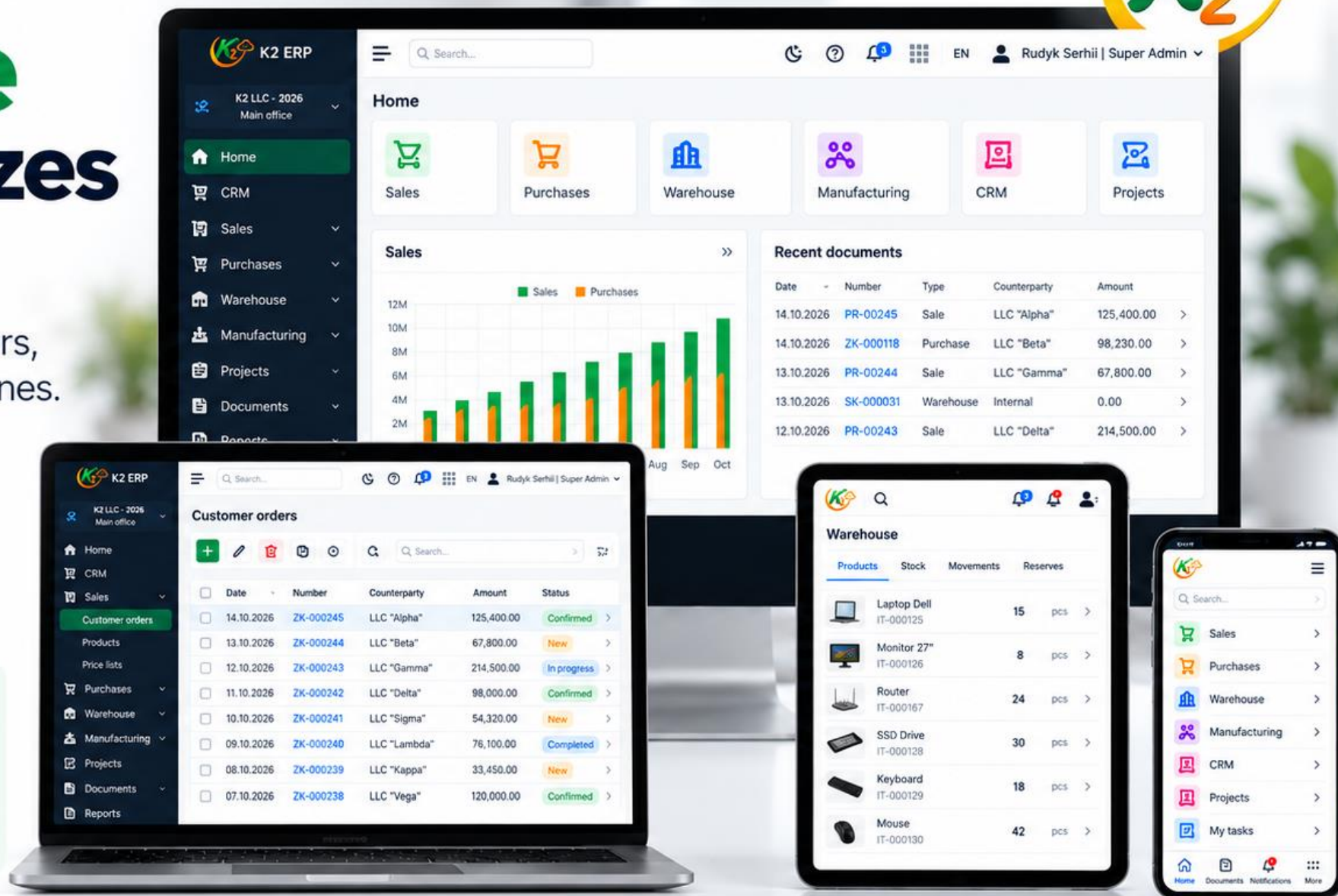
Web technologies allow you to create interfaces for monitors, laptops, tablets and smartphones.

The interface automatically adapts to the implementation of a specific component.



Advantage:

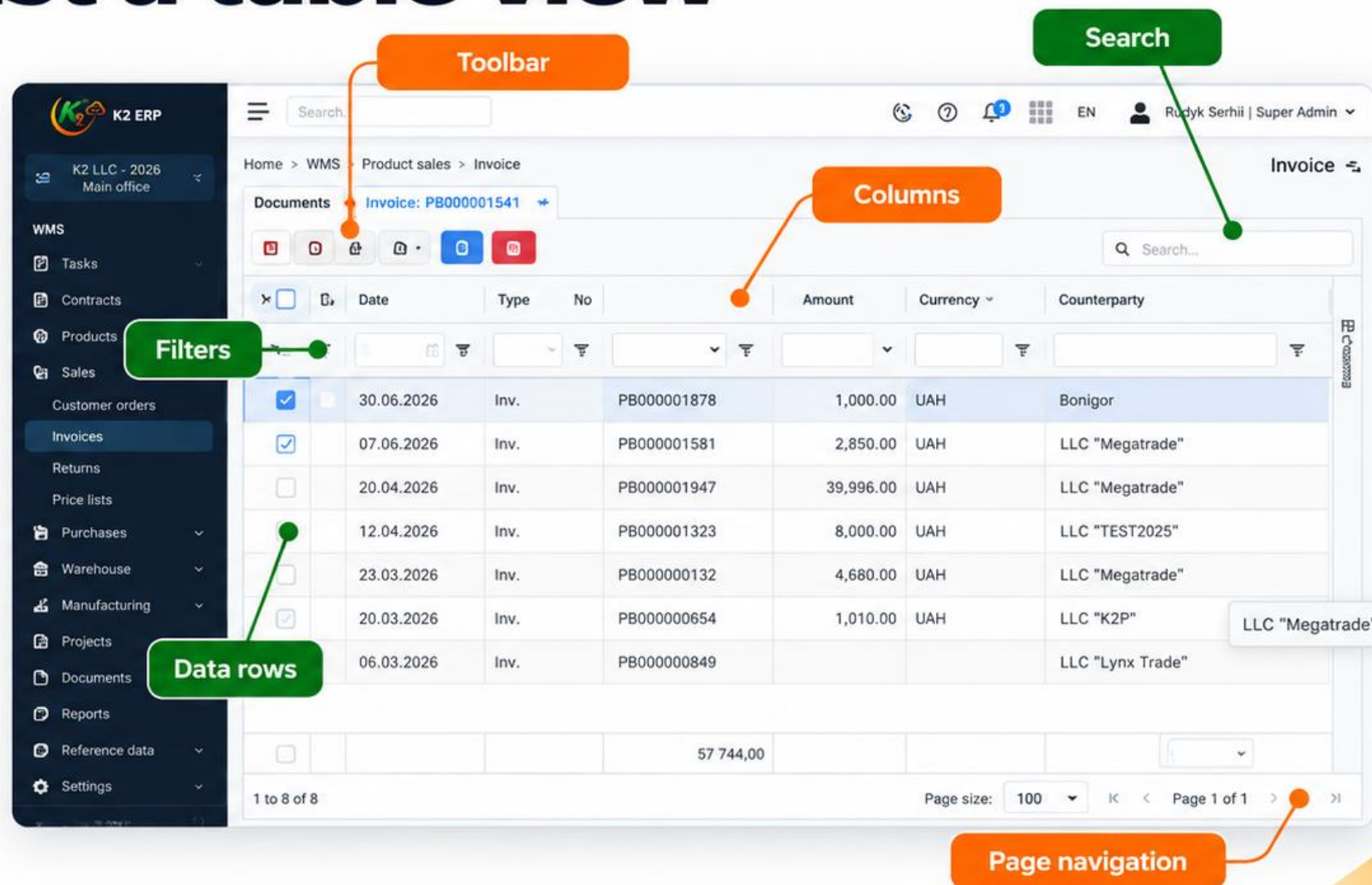
Ability to implement different user scenarios on a single platform.



Grids: not just a table view

K2 table components support operations with corporate data: browsing, sorting, search, filtering and other built-in tools.

The interface helps you not only see the data, but also work with it.



Toolbar

Search

Columns

Filters

Data rows

Page navigation

	Date	Type	No	Amount	Currency	Counterparty
☒	30.06.2026	Inv.	PB000001878	1,000.00	UAH	Bonigor
☒	07.06.2026	Inv.	PB000001581	2,850.00	UAH	LLC "Megatrade"
☐	20.04.2026	Inv.	PB000001947	39,996.00	UAH	LLC "Megatrade"
☐	12.04.2026	Inv.	PB000001323	8,000.00	UAH	LLC "TEST2025"
☐	23.03.2026	Inv.	PB000000132	4,680.00	UAH	LLC "Megatrade"
☒	20.03.2026	Inv.	PB000000654	1,010.00	UAH	LLC "K2P"
☒	06.03.2026	Inv.	PB000000849			LLC "Lynx Trade"
				57 744,00		

Advantage:

a ready-made multifunctional component for working with data sets.

Different representations of the same data

A unified platform — different ways to work with information.

Tables and filters.

Analytical views.

Graphical processes.



Advantage:

Different ways to work with the same data in one platform.

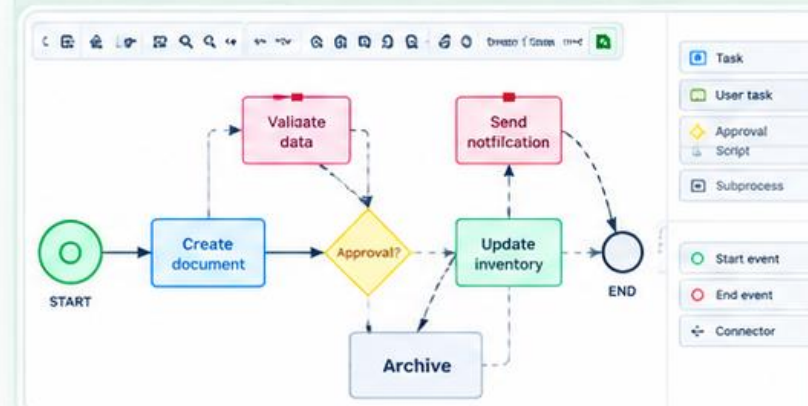
Table view — document journal

Deleted	Doc	Date	Type	No	Amount	Currency	Counterparty
☒		30.06.2026	Inv.	PB000001678	1,000.00	UAH	Bonigor
☒		07.06.2026	Inv.	PB000001591	2,850.00	UAH	LLC "Megatrade"
☐		20.04.2026	Inv.	PB000001541	39,996.00	UAH	LLC "Megatrade"
☐		12.04.2026	Inv.	PB000001497		UAH	
☒		23.03.2026	Inv.	PB000001323	8,000.00	UAH	LLC "TEST2025"
☐		20.03.2026	Inv.	PB000001312	4,888.00	UAH	LLC "K2P"
☐		09.03.2026	Inv.	PB000000854	1,010.00	UAH	LLC "Lynx Trade"

Analytical view — Pivot Grid

	Quantity	Amount (UAH)
> Electronics		
> Laptops	120	2 540 000
Monitors	85	680 000
> Accessories	240	320 000
> Office Equipment	110	410 000
> Peripherals	65	265 000
Total	620	4 215 000

Graphical view — business process diagram



Custom action handlers

A standard component can be extended with custom actions and business logic.

Example: validating an order, calculating the total amount, changing the document status or creating a related document.



Advantage:

Extend standard mechanisms with order-specific business logic.



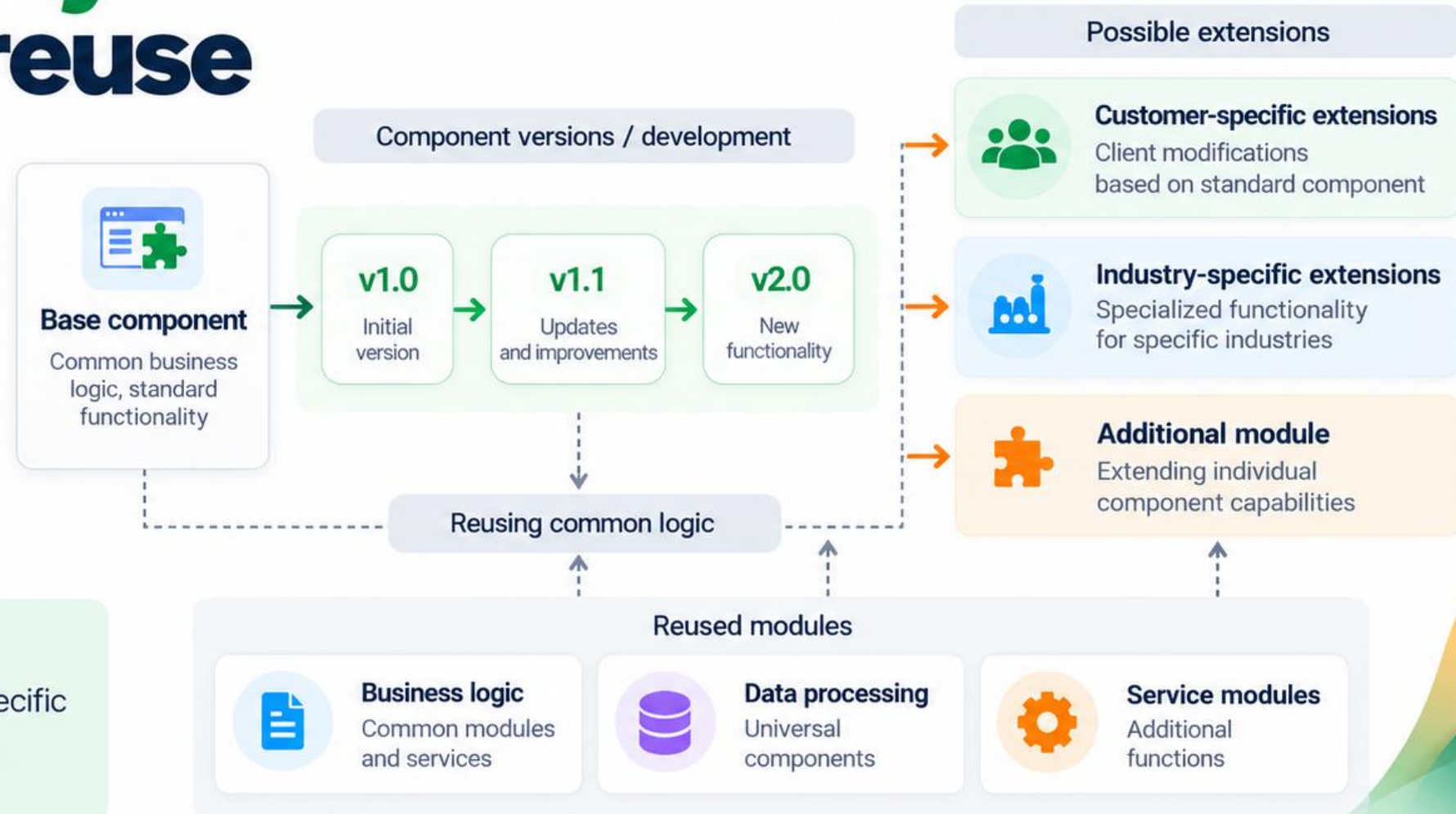
The screenshot displays the K2 ERP interface. On the left is a sidebar with a menu including Home, CRM, Sales, Customer orders (highlighted), Products, Invoices, Returns, Purchases, Warehouse, Manufacturing, Finance, Directories, Reports, and Settings. The main area shows the 'Customer orders (edit)' form. A red box highlights the 'Validate' button in the top toolbar, with a red arrow pointing to a code editor window. The code editor shows the following Python code:

```
1 def action_check_order(self, doc):
2     # Check counterparty
3     if not doc.partner_id:
4         self.ui.message('Counterparty is not specified')
5         return
6
7     # Calculate total amount
8     total = 0
9     for row in doc.rows:
10        total += row.quantity * row.price
11    if total <= 0:
12        self.ui.message('Amount cannot be zero')
13        return
14
15    # Change document status
16    doc.sum = total
17    doc.state = 'verified'
18    doc.save()
19    self.ui.message('Order validated')
```

Below the code editor, a modal dialog box titled 'K2 ERP' displays a green checkmark and the text 'Order validated', with an 'OK' button. In the background, the 'Customer orders (edit)' form is visible, showing fields for Number (ZK-000244), Date (13.10.2026), Counterparty (LLC "Beta"), Contract (Main contract), Amount (UAH) (98 230,00), Warehouse (Main warehouse), Responsible (Rudyk Serhii), and Comment. A table at the bottom shows a list of orders with columns for amount and status, with the first row highlighted in green and labeled 'Verified'.

Extensibility and code reuse

Modules and components can be adapted, extended and reused across different architectures and licensing environments.



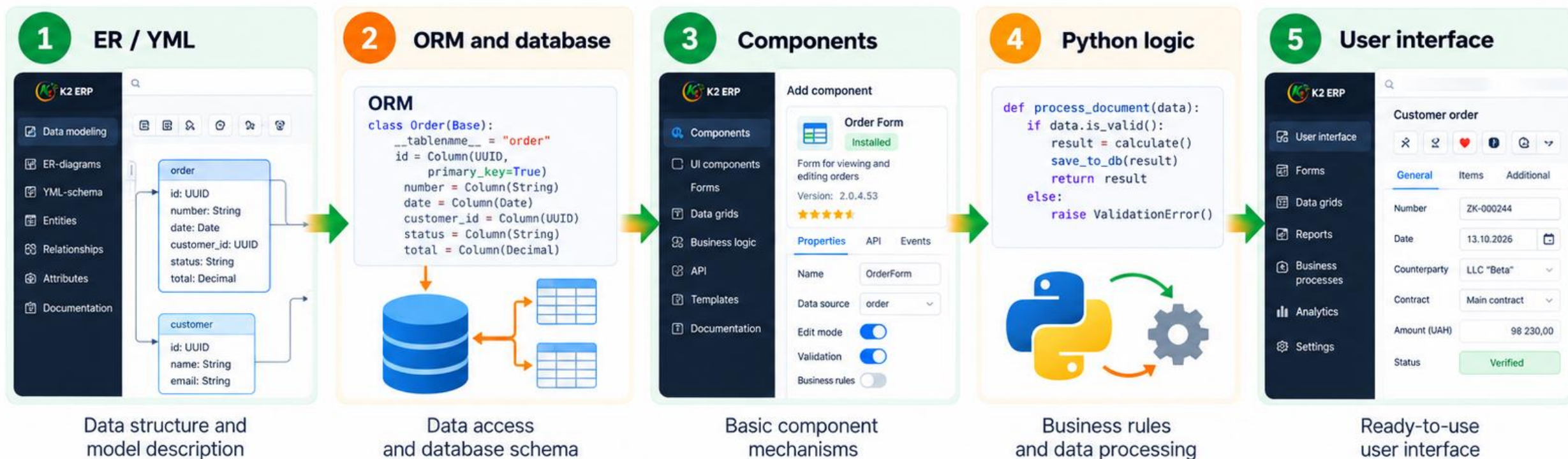
Advantage:

Supports industry-specific and client-specific customizations.

From data model to a working application

ER/YML → ORM and database → components → Python logic → user interface.

K2 unites these stages in a single technological process.

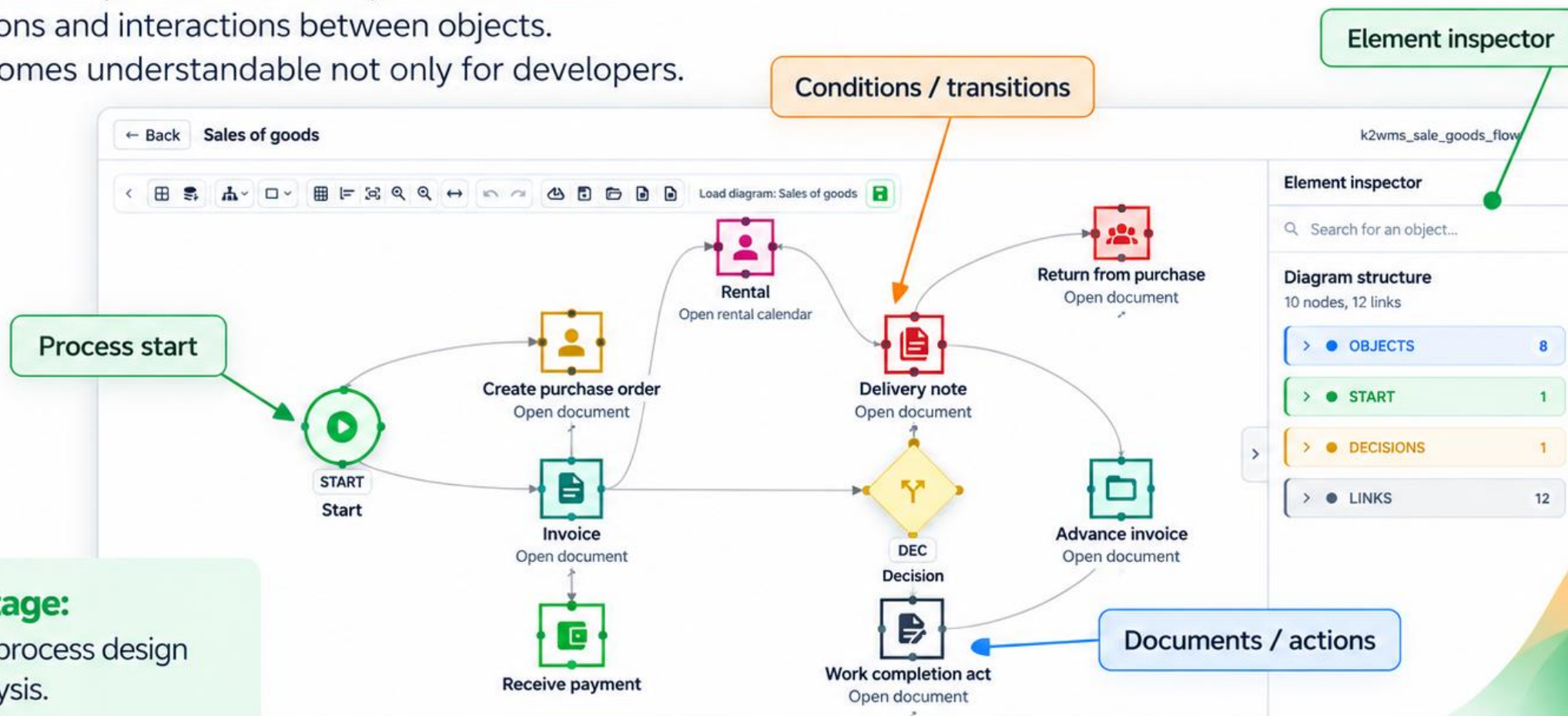


Advantage:

A consistent model for creating and developing an application.

K2 Process Editor: business processes in a visual way

Visual process models help describe the sequence of actions, conditions, transitions and interactions between objects.
Business logic becomes understandable not only for developers.



Advantage:

intuitive process design
and analysis.

BPM model and program code

The model describes the process. Python implements specific calculations and handlers where they are needed. Together, they create application logic.

K2 ERP

- Home
- CRM
- Customer orders**
- Products
- Invoices
- Returns
- Purchases
- Warehouse
- Manufacturing
- Finance
- Directories
- Reports
- Settings

Customer orders - BPM

Save Validate Publish Export Import

Elements

- Start
- End
- User task
- Decision
- Process
- Subprocess
- Page
- Timer
- Message
- Document
- Data source
- Group
- Text

```

graph TD
    Start((Start)) --> Validate[Validate data]
    Validate --> Decision{Amount > 0?}
    Decision -- No --> ShowError[Show error]
    ShowError --> End1((End))
    Decision -- Yes --> Calculate[Calculate total]
    Calculate --> CreateDoc[Create document]
    CreateDoc --> ChangeStatus[Change status]
    ChangeStatus --> End2((End))
  
```

Properties

Name: Calculate total

Type: Service task

Handler: calc_order_total

Description: Calculate the order total with discounts and taxes based on cart items.

Asynchronous execution

Continue on error

Input parameters

Output parameters

Additional

handlers.py

```

1 def calc_order_total(self, doc):
2     """Calculate the total order amount"""
3     total = 0
4
5     for row in doc.rows:
6         qty = row.quantity or 0
7         price = row.price or 0
8         total += qty * price
9
10    # Apply discount
11    discount = doc.discount or 0
12    if discount:
13        total -= total * discount / 100
14
15    # Apply tax
16    vat = doc.vat_rate or 0
17    if vat:
18        total += total * vat / 100
19
20    doc.sum = round(total, 2)
21    doc.save()
22    self.ui.message(f'Order total: {doc.sum:.2f}')
23    return doc.sum
  
```



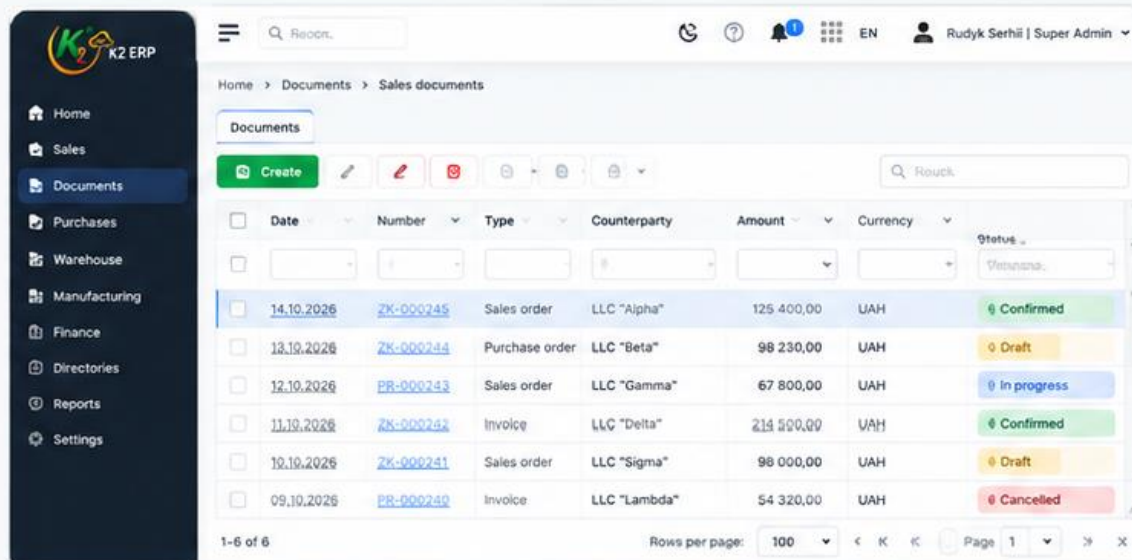
Advantage:

Combining process design with specialized logic implementation.

K2 Document Mechanism

Create documents, manage their details, tabular parts, related objects and printed forms.
A ready-made foundation for real business documents.

Document journal

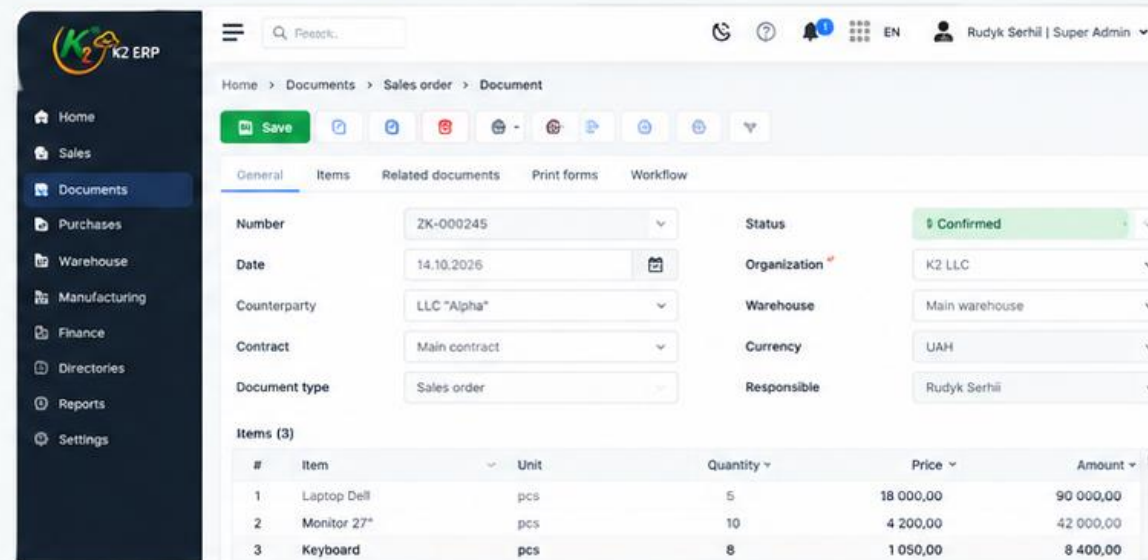


The screenshot shows the 'Document journal' interface in K2 ERP. It features a sidebar with navigation options: Home, Sales, Documents (selected), Purchases, Warehouse, Manufacturing, Finance, Directories, Reports, and Settings. The main area displays a list of documents under the 'Documents' tab. The list includes columns for Date, Number, Type, Counterparty, Amount, Currency, and Status. The status column shows various states like 'Confirmed', 'Draft', 'In progress', and 'Cancelled'.

Date	Number	Type	Counterparty	Amount	Currency	Status
14.10.2026	ZK-000245	Sales order	LLC "Alpha"	125 400,00	UAH	Confirmed
13.10.2026	ZK-000244	Purchase order	LLC "Beta"	98 230,00	UAH	Draft
12.10.2026	PR-000243	Sales order	LLC "Gamma"	67 800,00	UAH	In progress
11.10.2026	ZK-000242	Invoice	LLC "Delta"	214 500,00	UAH	Confirmed
10.10.2026	ZK-000241	Sales order	LLC "Sigma"	98 000,00	UAH	Draft
09.10.2026	PR-000240	Invoice	LLC "Lambda"	54 320,00	UAH	Cancelled

Document list, search and filters

Document card



The screenshot shows the 'Document card' interface in K2 ERP. It features a sidebar with navigation options: Home, Sales, Documents (selected), Purchases, Warehouse, Manufacturing, Finance, Directories, Reports, and Settings. The main area displays the details of a specific document (Sales order ZK-000245). The details are organized into sections: General, Items, Related documents, Print forms, and Workflow. The 'General' section includes fields for Number, Date, Counterparty, Contract, Document type, Status, Organization, Warehouse, Currency, and Responsible. The 'Items' section displays a table of items with columns for #, Item, Unit, Quantity, Price, and Amount.

#	Item	Unit	Quantity	Price	Amount
1	Laptop Dell	pcs	5	18 000,00	90 000,00
2	Monitor 27"	pcs	10	4 200,00	42 000,00
3	Keyboard	pcs	8	1 050,00	8 400,00

Document details, counterparty, attachments and tabular part



Advantage:

Reusable document structure for various business scenarios.

Document creation based on other objects

One document can become the basis for the next step in a business process.
For example, a customer order is linked to subsequent operational documents.



Customer order

Process start.
Capturing customer requirements.



Goods delivery

Shipment of goods to the customer based on the order.



Customer invoice

Creating an invoice based on delivery or order.



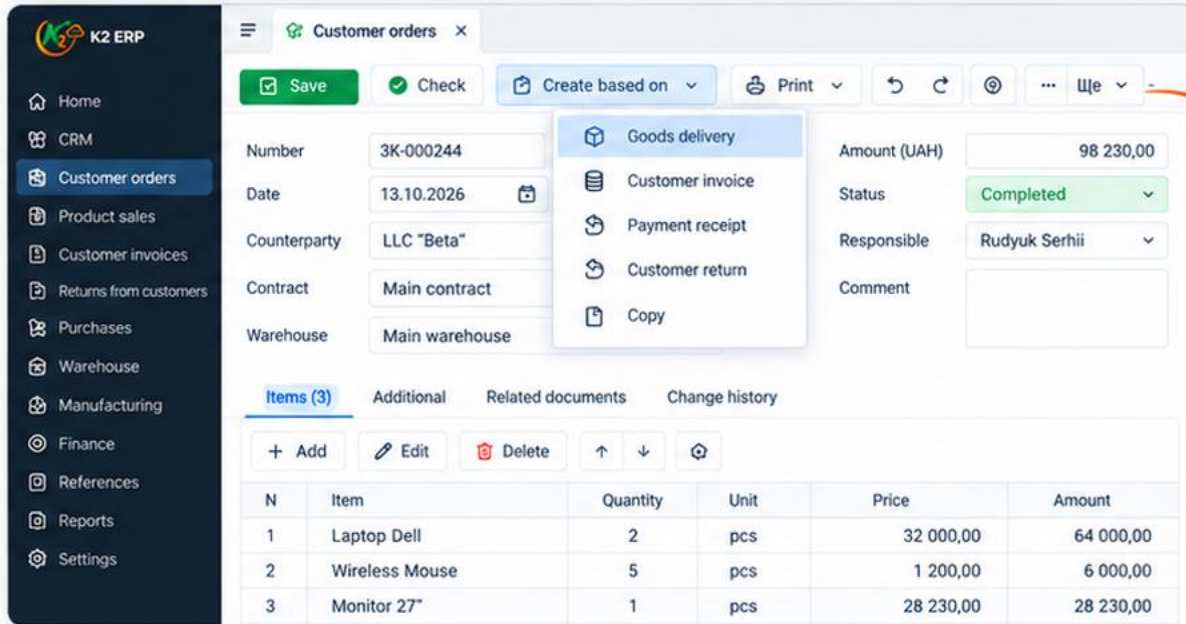
Payment receipt

Registering payment from the customer.



Customer return

If needed — return of goods based on delivery.



K2 ERP Customer orders

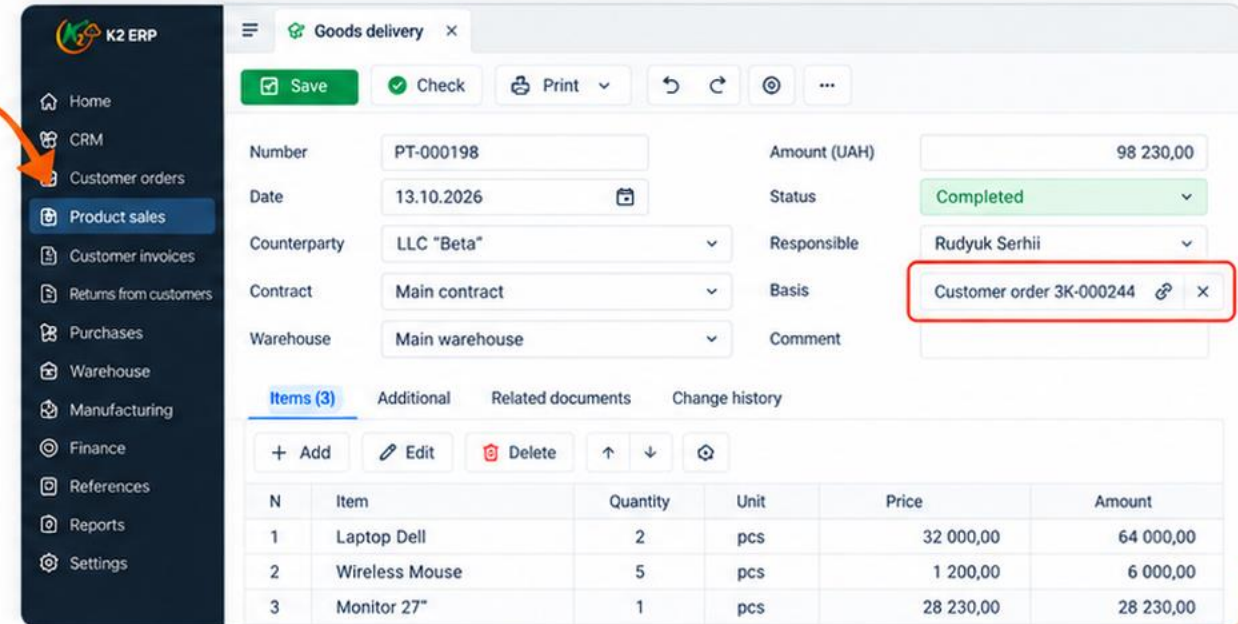
Save Check Create based on Print

Number: 3K-000244
Date: 13.10.2026
Counterparty: LLC "Beta"
Contract: Main contract
Warehouse: Main warehouse

Amount (UAH): 98 230,00
Status: Completed
Responsible: Rudyuk Serhii

Items (3)

N	Item	Quantity	Unit	Price	Amount
1	Laptop Dell	2	pcs	32 000,00	64 000,00
2	Wireless Mouse	5	pcs	1 200,00	6 000,00
3	Monitor 27"	1	pcs	28 230,00	28 230,00



K2 ERP Goods delivery

Save Check Print

Number: PT-000198
Date: 13.10.2026
Counterparty: LLC "Beta"
Contract: Main contract
Warehouse: Main warehouse

Amount (UAH): 98 230,00
Status: Completed
Responsible: Rudyuk Serhii

Basis: Customer order 3K-000244

Items (3)

N	Item	Quantity	Unit	Price	Amount
1	Laptop Dell	2	pcs	32 000,00	64 000,00
2	Wireless Mouse	5	pcs	1 200,00	6 000,00
3	Monitor 27"	1	pcs	28 230,00	28 230,00



Advantage:

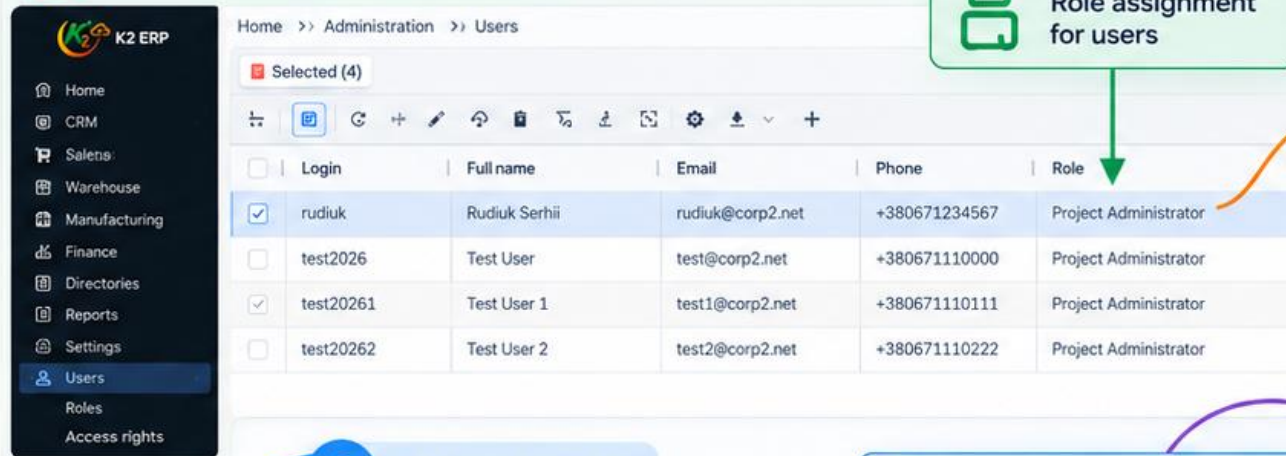
Building interconnected document-based business processes.

Users, roles and permissions

Access management is part of the corporate platform.

Permissions define which actions and information are available to each specific user.

1 Users and roles



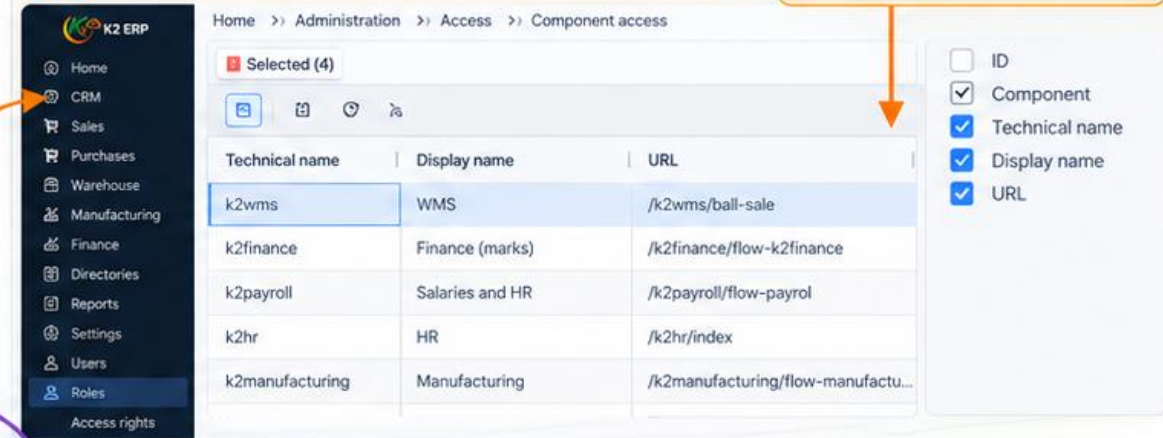
Home >> Administration >> Users

Selected (4)

	Login	Full name	Email	Phone	Role
☒	rudiuk	Rudiuk Serhii	rudiuk@corp2.net	+380671234567	Project Administrator
☐	test2026	Test User	test@corp2.net	+380671110000	Project Administrator
☒	test20261	Test User 1	test1@corp2.net	+380671110111	Project Administrator
☐	test20262	Test User 2	test2@corp2.net	+380671110222	Project Administrator

Role assignment for users

2 Roles and access to components



Home >> Administration >> Access >> Component access

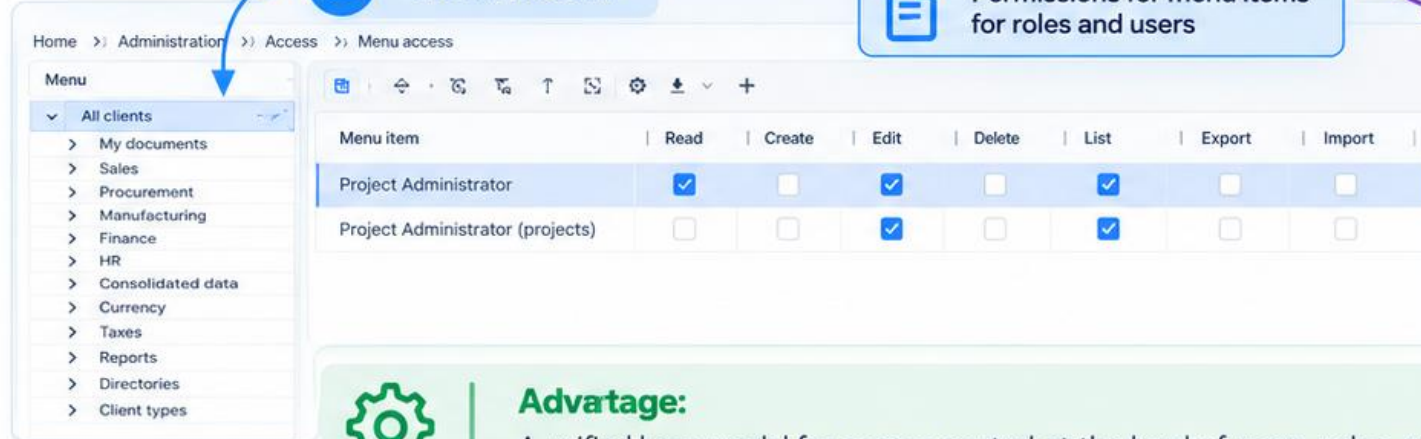
Selected (4)

Technical name	Display name	URL
k2wms	WMS	/k2wms/ball-sale
k2finance	Finance (marks)	/k2finance/flow-k2finance
k2payroll	Salaries and HR	/k2payroll/flow-payrol
k2hr	HR	/k2hr/index
k2manufacturing	Manufacturing	/k2manufacturing/flow-manufactu...

Access to system components (modules)

- ☐ ID
- ☒ Component
- ☒ Technical name
- ☒ Display name
- ☒ URL

3 Menu access



Home >> Administration >> Access >> Menu access

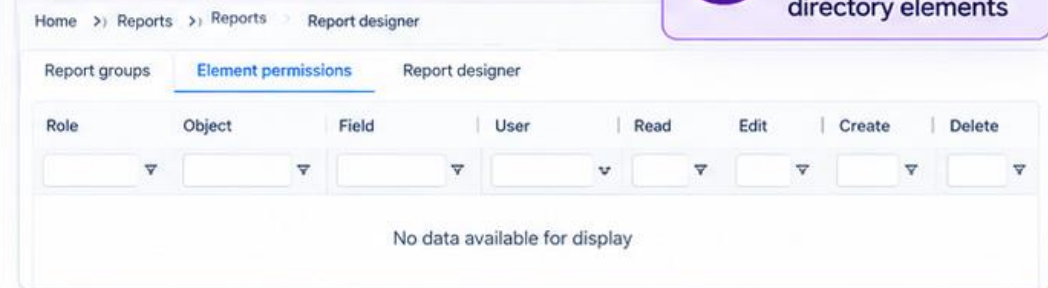
Menu

- ▼ All clients
 - > My documents
 - > Sales
 - > Procurement
 - > Manufacturing
 - > Finance
 - > HR
 - > Consolidated data
 - > Currency
 - > Taxes
 - > Reports
 - > Directories
 - > Client types

Permissions for menu items for roles and users

Menu item	Read	Create	Edit	Delete	List	Export	Import
Project Administrator	☒	☐	☒	☐	☒	☐	☐
Project Administrator (projects)	☐	☐	☒	☐	☒	☐	☐

4 Data element permissions



Home >> Reports >> Reports >> Report designer

Report groups | Element permissions | Report designer

Role	Object	Field	User	Read	Edit	Create	Delete

No data available for display

Restricting access to specific data and directory elements



Advantage:

A unified base model for access control at the level of users, roles, menus, components and data.

Organizational structure

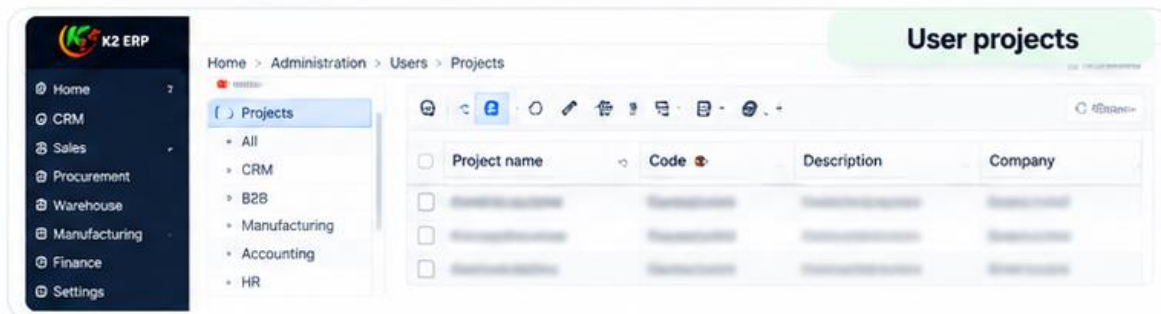
Users, companies, departments and roles can be combined into appropriate organizational models.

This is important for systems that operate within a single office or across multiple domains.



Advantage:

support for corporate structures with role-based access control.



Data, documents, processes and users

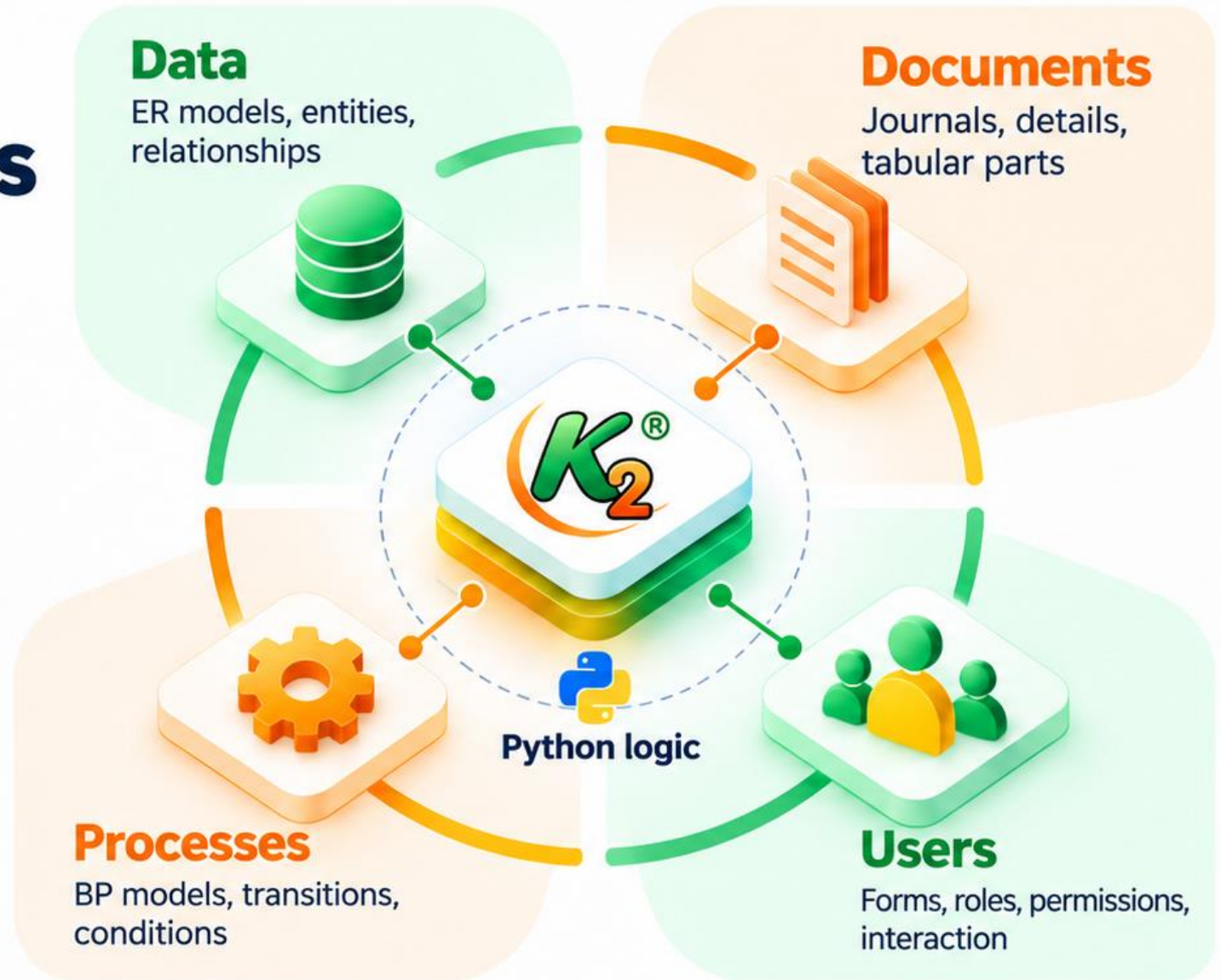
ER describes data, BP — processes.
Python — specialized logic.
K2 components enable interaction
with the user.

These mechanisms work as parts
of a single platform.



Advantage:

A unified approach to building
a corporate application.



K2 report designer

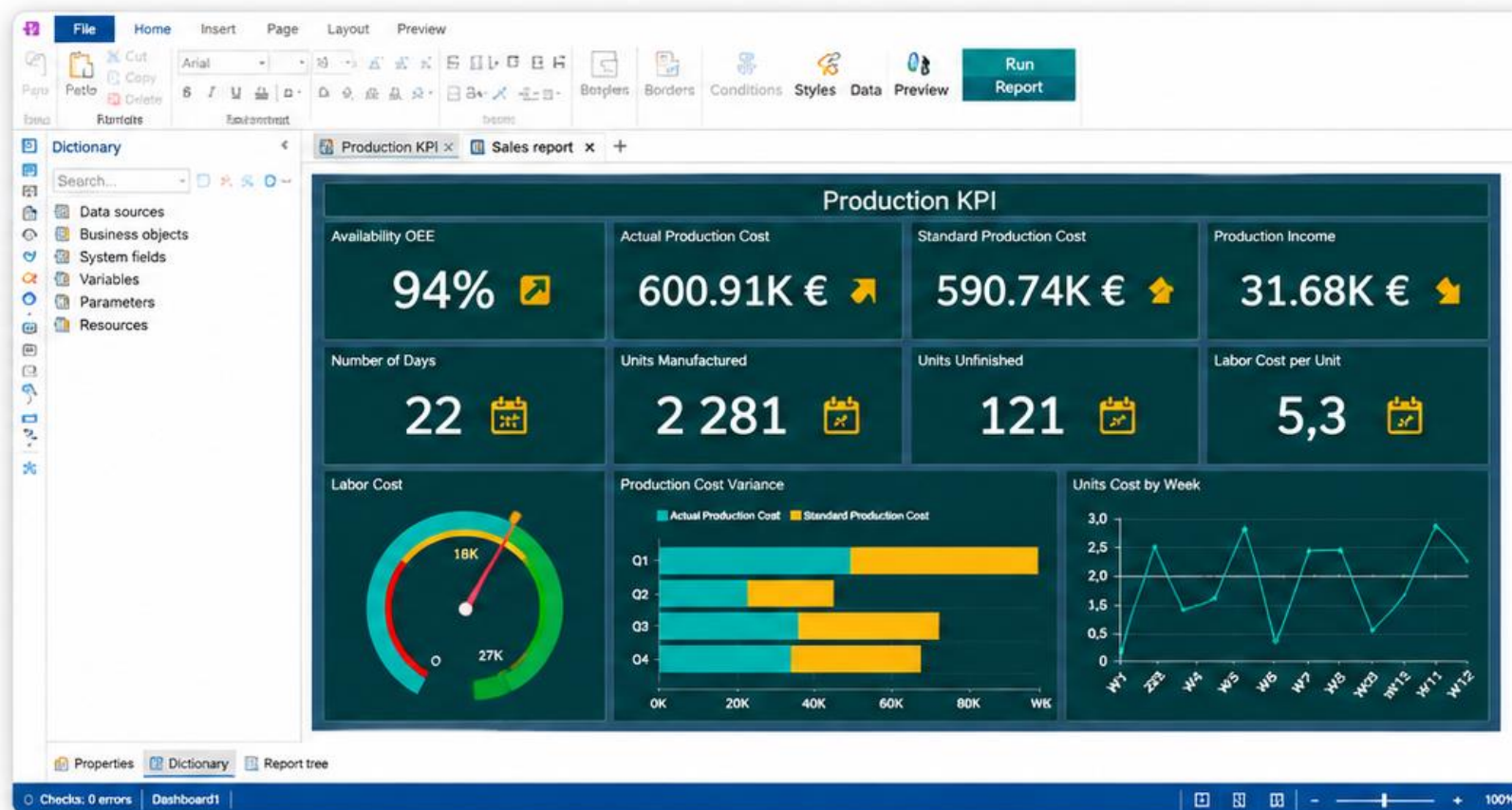
Data, filters, grouping, calculations and result presentation — in one tool.

K2 helps you create reports without developing each of them as a separate application.



Advantage:

Less typical report development.



Data sources

Report workspace

Visual representation

Printed forms and templates

Invoices, waybills, acts, reports and other printed documents can be created using a corresponding designer.

K2 allows you to reuse document preparation tools without developing a separate application for each template.



Printed form designer



Advantage

Reuse document preparation tools.



Result / ready-to-print report

PivotGrid: analytics without a new module

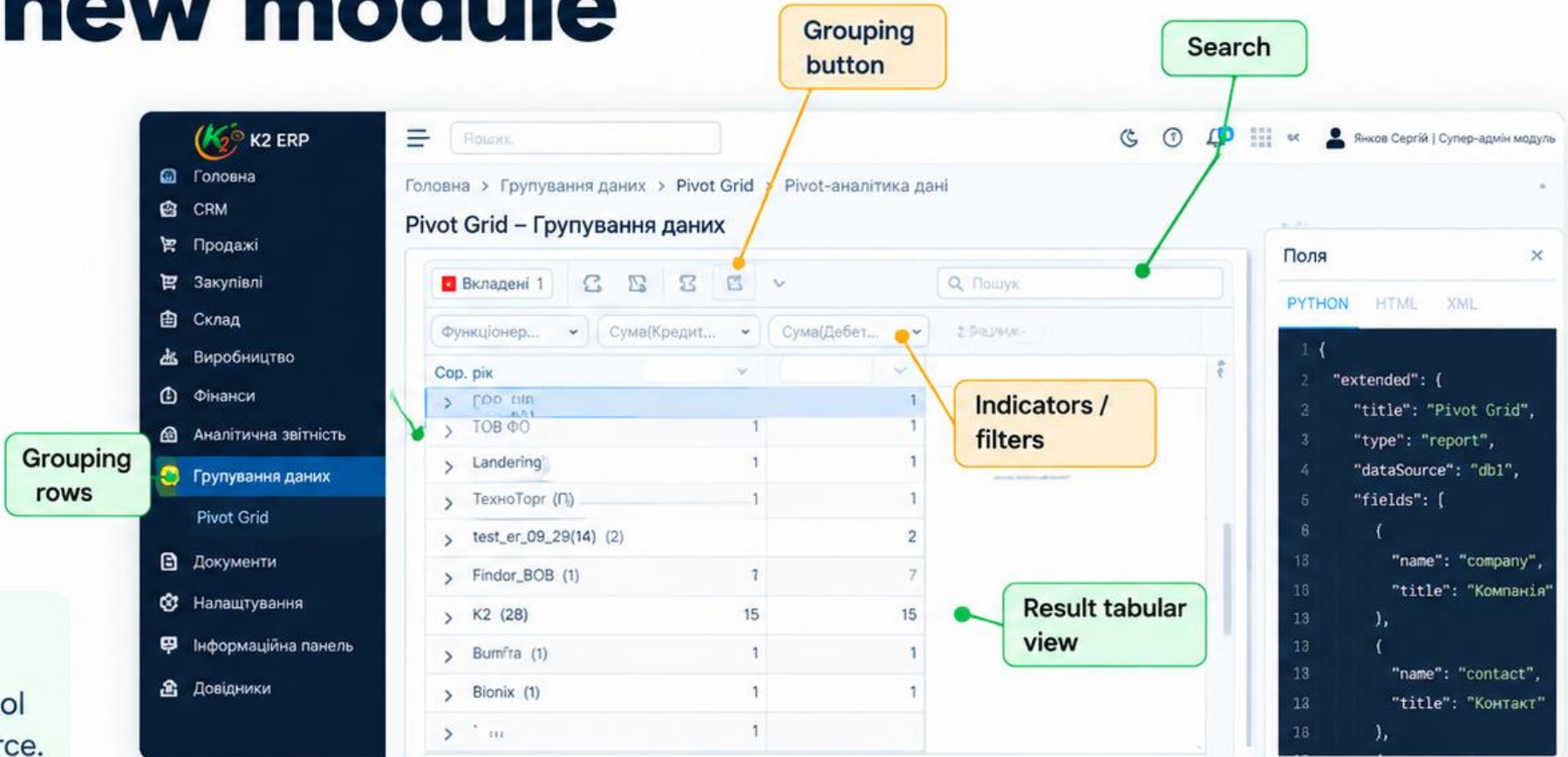
Make data-driven decisions by grouping data, analyzing it and visualizing the results in the form of indicators.

Work with the data of the new module by default with its enabled.



Advantage:

A flexible analytical tool based on the data source.



The screenshot shows the K2 ERP interface with the Pivot Grid module selected in the sidebar. The main area displays a table of data with columns for 'Сор. рік' (Year) and 'Сума' (Sum). Annotations highlight key features:

- Grouping rows:** Points to the 'Сор. рік' column header.
- Grouping button:** Points to the 'Групування' button in the top toolbar.
- Search:** Points to the search bar in the top right corner.
- Indicators / filters:** Points to the 'Сума(Кредит...)' and 'Сума(Дебет...)' dropdown menus.
- Result tabular view:** Points to the table displaying data rows.

The table data is as follows:

Сор. рік	Сума(Кредит...)	Сума(Дебет...)
> ТОВ ФО	1	1
> Landering	1	1
> ТехноТорг (П)	1	1
> test_er_09_29(14) (2)		2
> Findor_BOB (1)	1	7
> K2 (28)	15	15
> Bum'ra (1)	1	1
> Bionix (1)	1	1
> ...	1	

On the right, a 'Поля' (Fields) panel shows a JSON configuration for the Pivot Grid report:

```

1 {
2   "extended": {
3     "title": "Pivot Grid",
4     "type": "report",
5     "dataSource": "db1",
6     "fields": {
7       {
8         "name": "company",
9         "title": "Компанія"
10      },
11      {
12         "name": "contact",
13         "title": "Контакт"
14      }
15    }
16  }
17 }
  
```

OLAP and multidimensional analytics

Analyse data by periods, departments, product groups, counterparties and other available dimensions.

Interactive analytics based on your system data without developing a separate module.



BI: from transactions to insights

Operational data → management metrics, charts and analytical dashboards



Advantage:

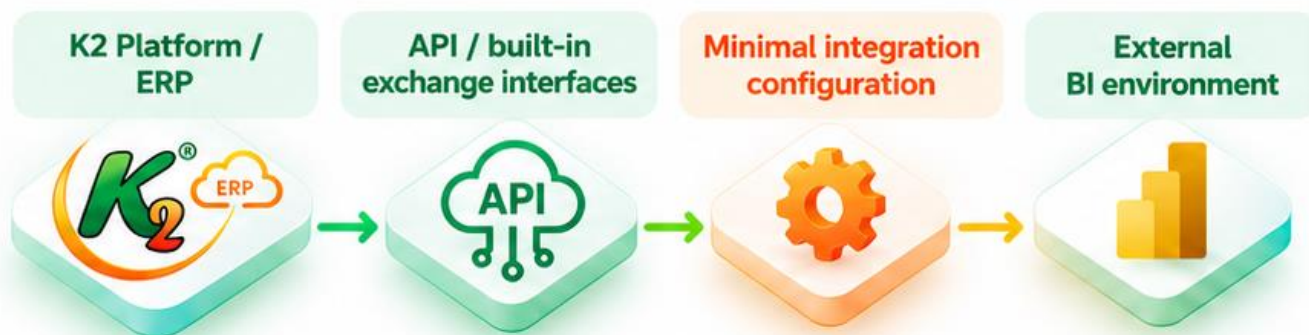
Combines operational accounting and analytics.



K2 data in external analytics tools



API and built-in interfaces for data exchange enable you to use K2 data in external BI infrastructures. Integration requires only minimal configuration of access, data exchange and the analytical model.

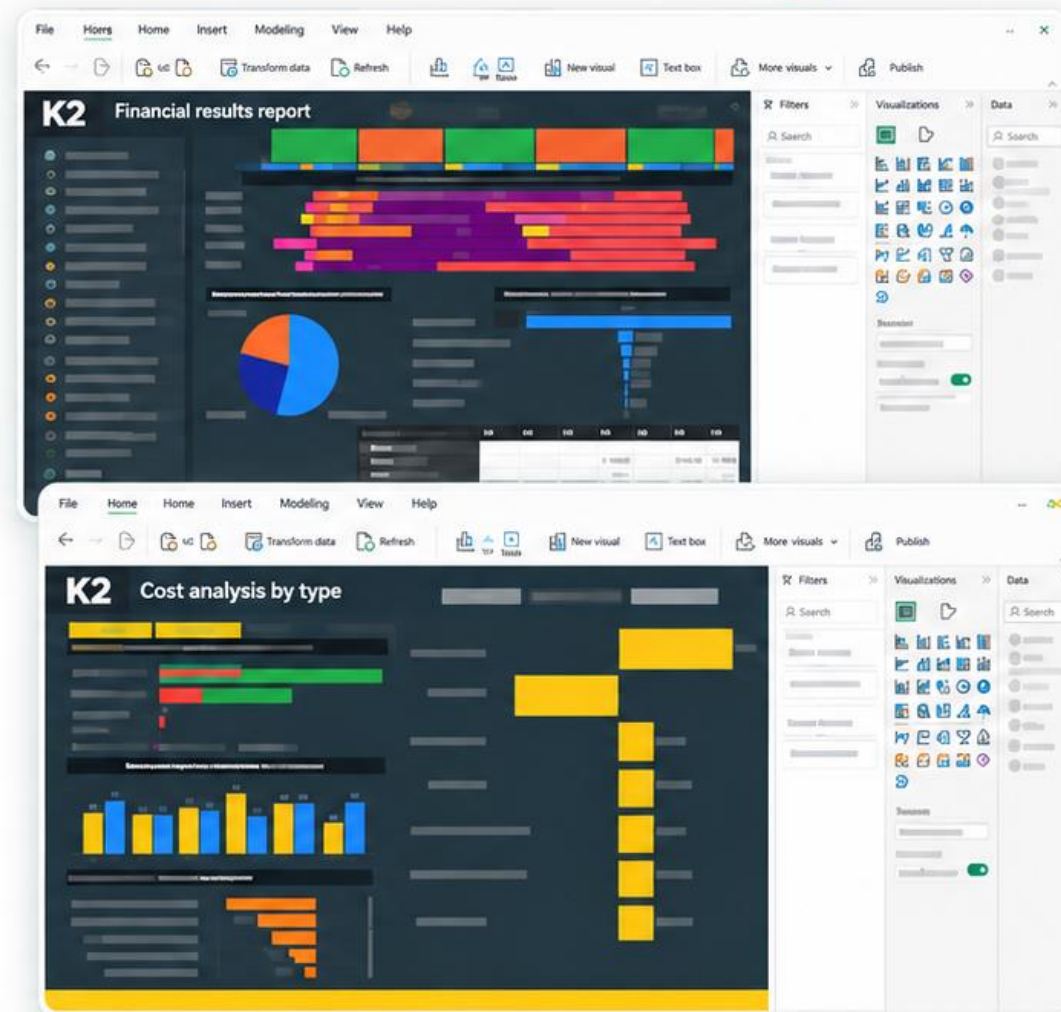


External analytics works through integration, not as a built-in K2 module.



Advantage:

You can connect K2 data to existing analytics tools.



Reports as reusable components

Ready-made reports, templates and analytical elements can be **adapted** for future business tasks and clients within the **K2 platform**.



Knowledge accumulation

Your own repository of reports and templates.



Reuse

Adaptation of existing reports for new tasks and clients.



Unified standards

Common styles, elements and approaches to report creation.



Quick adaptation

Reduced volume of custom development.



API and integrations

K2 can interact with external systems, services and applications.

APIs allow you to combine data and functionality in more complex software solutions.



Advantage:

integration of the platform into an existing IT ecosystem.

Website

Integration with a web portal, data transfer, process automation.



HTTPS
JSON
REST API

Mobile application

Mobile apps for working with data and business processes.



HTTPS
JSON
REST API

External service

Integration with external online services, data exchange.



HTTPS
JSON
REST API

Corporate system

Data exchange with ERP, CRM and other internal systems.



HTTPS
JSON
REST API



Exchange format: REST API / JSON / HTTPS

K2 Replicator

Data synchronization and transfer without tight coupling to a single system type



K2 Platform


Home
Replicator
Settings

1. Schema parameters
New and existing maps

2. XML structure
Source data filters

3. Field mapping
Mapping to K2 tables

4. Readiness check
What will be started

5. Start
Go to import

Save schema Save as template To start Draft Save and exit Back to tree Advanced settings

2. XML structure

XML structure tree, mapping of nodes and fields. Select a node and a field to set the mapping.

Search by XML structure, table or field name Clear search

All (164) Unmapped (101) Partially (0) Mapped (63)

With errors (164) Documents (29) Reference data (59) Plans and calculators (1) Registers (5) Other (70)

With schema/XML (164) New in XML (164) Already in system (0)

With data (164) Empty (0) Not empty (164)

Expand all Collapse all

<CatalogObject.BankStatement>
table | CatalogObject.BankStatement | mapped 10/15 | children: 15
Field mapped for read (15). Select a field to edit mapping.
Open mapping

<CatalogObject.Bank>
table | CatalogObject.Bank | mapped 7/12 | children: 12
Field mapped for read (15). Select a field to edit mapping.
Open mapping

<CatalogObject.CounterpartyContract>
table | CatalogObject.CounterpartyContract | mapped 8/49 | children: 49
Field mapped for read (15). Select a field to edit mapping.
Open mapping


Home
Replicator
Browse
Data sources
Schema templates
Mapping templates
Field mapping
Launches
Log
Settings

Start import Cancel import Stop import Show last errors Go to schema settings

For subsequent launches, the organization_id field must be specified, to which K2 documents will be associated.

Import log

```
[18:46:54] documentobject_vouchersdenymyglanx_andyheenzzyahuderr@laozyah: not found LineObject(1) in K2
[18:46:55] Index: success
[18:46:55] documentobject_zayavlenyepogooeyplbz: not found LineObject(1) in K2
[18:46:58] informationregisterrecords#_scheta_abncls_tarumetodtriy: not found LineObject(1) in K2
[18:46:58] informationregisterrecords#_tseny_nomenklatury: not found LineObject(1) in K2
[18:46:58] Import completed
[18:46:58] XML structure: 105, imported: 58, partially: 1, skipped: 46, errors: 0, duration: 9 minutes; application: 760219/760234
```

Total items
105

Imported
58

Partially
1

Skipped
46

Errors
0

With links
0

Application used
760219/760234

Edit schema To start To completion

Statistics by structures

Structure	Status	Rows	Message
k2requisites	success	4/4	
k2bank	success	4/4	



Legacy systems /
Databases /
XML / Other solutions

Mapping rules
Synchronization
Import control



K2 Replicator



K2 Platform



Works with various
data sources,
not just 1C/BAS.



Advantage:

Step-by-step and controlled data exchange between different systems.



Developed in Ukraine

erp.kyiv.ua

77

K2 Update — lifecycle of components

From source code to controlled installation and updates of K2 components



K2 Update component catalog

Install components

Available: 89 Installed: 47 [Component catalog](#) Updates available until: 03.05.2027

Update history [Check for updates](#) 11

Search components...

Report Designer K2report Installed

Current: 2.0.4.53 Latest: 2.0.4.53 ★★★★★ Comments: 5

Module for designing printed form templates and reports.

[Installed](#) [Component page](#)

Update Server, Marketplace K2update Available for installation

Stable: 2.0.1.31 License required ★★★★★ Comments: 2

K2 Cloud ERP Marketplace – a modern marketplace with additional features to extend the functionality of your ERP system. Our platform provides unique...

[Install](#) [Component page](#)

Test Component K2test Available for installation

Stable: 2.0.0.0 License required ★★★★★ Comments: 9

Example of a starter component created in Python.

[Install](#) [Component page](#)



Advantage:

Controlled development and reuse of components.



Developed in Ukraine

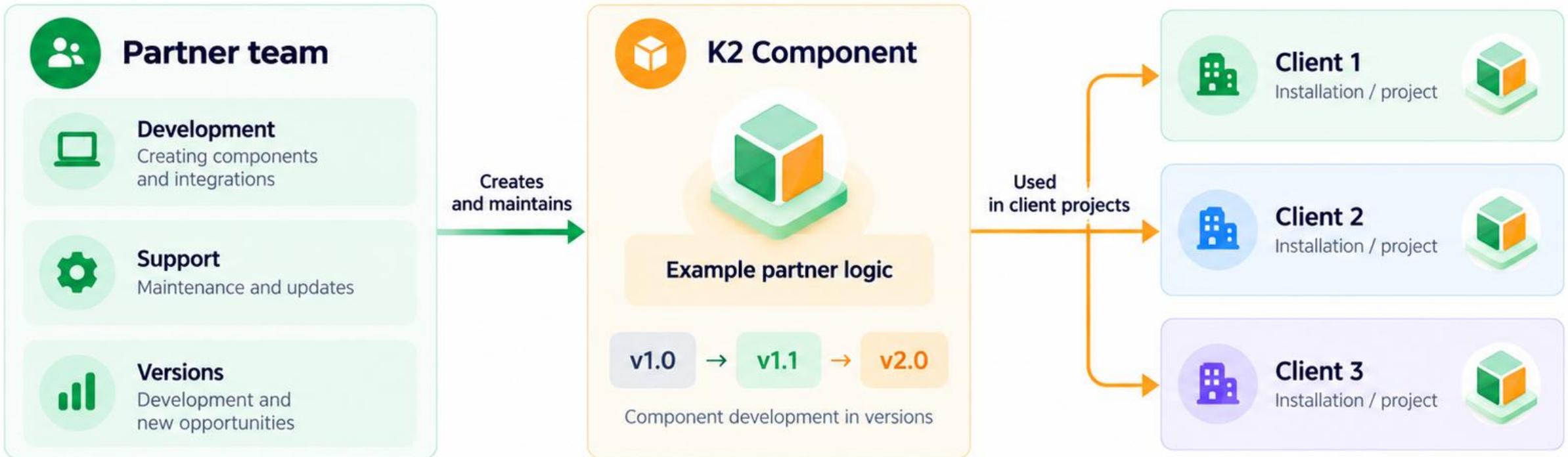
erp.kyiv.ua

Partner components



K2 Platform

Partners can create their own industry-specific modules and integrations, support them and use them in their own projects.



Advantage:

the opportunity to build your own product business on the platform.



Cloud or own server — you decide

Public cloud. Private cloud. Own server infrastructure. Hybrid deployment.
The deployment architecture is chosen according to your project requirements.



Public cloud

Deployment in external cloud infrastructure



Users
(office / remote)



Application Database



Private cloud

Isolated environment under customer control



Users
(office / remote)



Access
control



Application Database

Private cloud infrastructure



Own server infrastructure

Local deployment in your infrastructure



Users
(office / remote)



Corporate
network



Application
servers



Database



Hybrid deployment

Combination of local and cloud resources



Users
(office / remote)



Corporate
network



Local infrastructure
(application / data)



Cloud services
(application / data)



Advantage:

Flexible deployment model.

Cross-platform capability



K2 Platform

Access via browser on Windows, Linux and macOS.
Mobile scenarios are available on Android and iOS.



Advantage:

Lower dependence
on a specific client OS.

Local and offline scenarios



K2 supports modern office and desktop scenarios, when stable communication with the network is not available. Local work and synchronization are possible components of the platform.

1

Local work

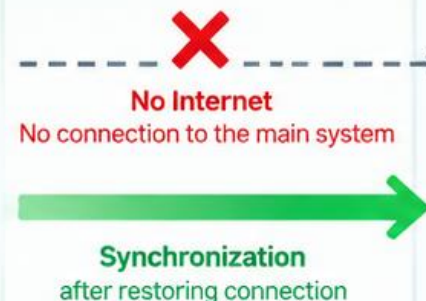
Cash desks, workplaces and local warehouses work without access to the Internet.



2

Offline usage & synchronization

After resuming connection, data is automatically synchronized with the central system in controlled mode.



Benefits:

Support for offline work processes and stable operation of the system.



Local workstation



Offline monitoring



Flexible scenarios



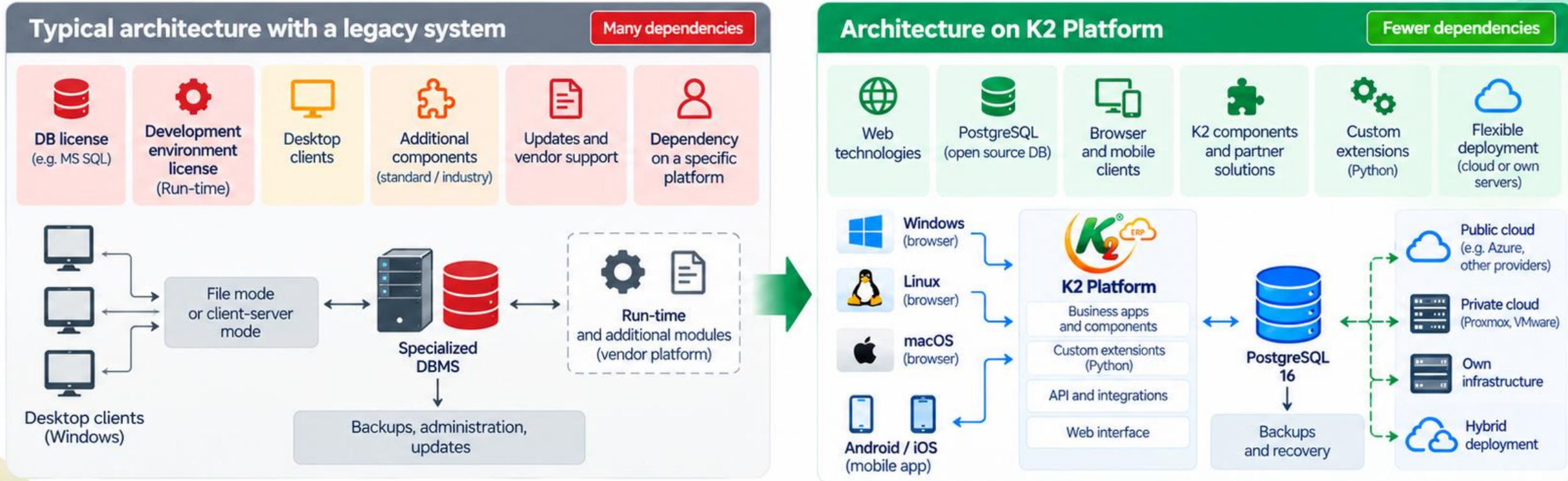
Developed in Ukraine

erp.kyiv.ua

82

Independence from legacy environments

After a complete replacement of a legacy system, the need for its specialized runtime, desktop clients and license dependencies can be eliminated. Costs for K2, infrastructure and third-party services are evaluated separately.



Benefits:

Reduced dependency on specific legacy platforms.



Less technological dependency

Use open and modern technologies.



Unified access from different devices

Browser, mobile and desktop clients.



Flexible deployment

Public cloud, private cloud, own servers or hybrid model.



Custom solutions and integrations

Extensions on Python, partner components.

Open source code and control over the result



Own infrastructure, accessible source code, modern technologies and modular approach.
The new product does not become an opaque black box.

K2 code structure

k2

> apps

> components

> core

> docs

> extensions

> frontend

> infra

> packages

> scripts

> services

> .gitignore

> docker-compose.yml

> README.md

> LICENSE

k2 > components > accounting

> api

> models

> reports

> services

> ui

> __init__.py

> manifest.json

> README.md

Git — version control

main

k2 / components / accounting

Serhii Rudiuk

Update accounting reports

2 days ago

Add VAT report

4 days ago

Fix document validation

1 week ago

Refactor services

1 week ago

Update models

2 weeks ago

Initial import

3 weeks ago

Latest commit a1b2c3d · main

PostgreSQL

PostgreSQL 16

Databases (4)

k2

Schemas (12)

accounting

crm

hr

manufacturing

public

reports

warehouse

Tables (86)

acc_document

acc_document_row

acc_cash

acc_counterparty

Views

Functions

Procedures

Query

Query History

1 SELECT * FROM acc_document

2 ORDER BY id DESC LIMIT 50;

Data

Messages

Notifications

	id	date	number
	bigint	date	character varying
1	154862	2026-10-14	PR-000125
2	154861	2026-10-14	PR-000124
3	154860	2026-10-13	PR-000123
4	154859	2026-10-13	PR-000122
5	154858	2026-10-12	PR-000121
6	154857	2026-10-12	PR-000120
7	154856	2026-10-11	PR-000119

Extensions and configuration

Home

Applications

Components

Repository

Updates

Installed

Configuration

Search...

Components

+ Create

Import

Name	Version
Accounting	1.4.2
Payroll	1.3.0
Warehouse	1.2.1
Manufacturing	1.2.0
CRM	1.5.1
Reports	1.1.3



Modular structure

Components, services, APIs, documentation.



Managed development

Change history, branches, rollback to previous versions.



Open technologies

PostgreSQL — reliable and widely used open source DB.



Flexible extensions

Custom modules, integrations and project-specific settings.



Technical control

Access to code and the ability to analyze and make changes.



Own infrastructure

Deployment on your own servers or in a selected cloud.



Modern technologies

Python, TypeScript, PostgreSQL, Nginx, Proxmox and others.



Modular approach

Each component is independent, manageable and open.



K2 is already used for real business tasks



Warehouse operations, document flow, reporting, access administration, data integration and other practical use cases.

WMS - Document Journal

Date	Type	No.	Amount	Counterparty
2024-05-20	Invoice	INV-000125	1,000.00 USD	LLC Megatrade
2024-05-18	Invoice	INV-000124	2,850.00 USD	LLC Interdealer
2024-05-16	Invoice	INV-000123	39,950.00 USD	LLC Test2025
2024-05-14	Invoice	INV-000122	8,000.00 USD	LLC Megatrade
2024-05-12	Invoice	INV-000122	4,880.00 USD	LLC LuxTrade

WMS – Document Journal

Documents – Document Card

Document No. INV-000125 Date 20.05.2024

Document type Invoice Organization LLC Megatrade

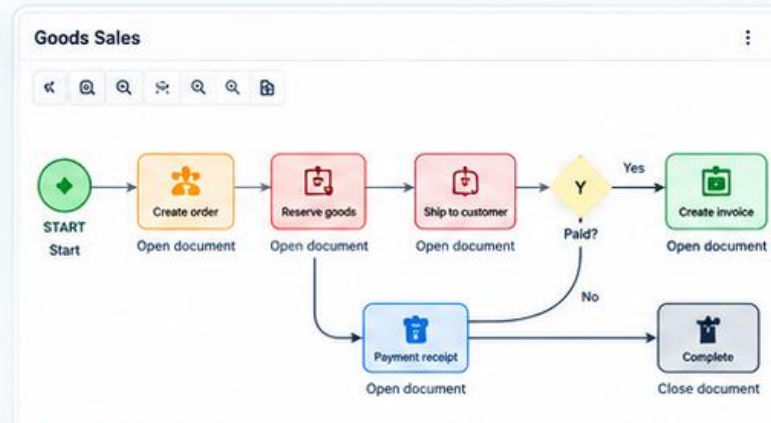
Counterparty LLC Interdealer Warehouse Main warehouse

Contract Select a contract... Counterparty contact Ivan Petrenko

Responsible Select a person... Currency USD

Comment Enter a comment...

Documents – Document Card



Goods Sales Process Flow



K2 Report Designer

Access and Roles

Menu	URL	View	Create	Edit	Delete	Export
All clients	/clients	✓	✓	✓	✓	✓
My documents	/docs	✓	✓		✓	✓
Reports	/reports	✓	✓	✓	✓	✓
Warehouses	/warehouse	✓	✓			
Bank	/bank	✓				
Exchange rates	/rates					
Currency	/rates					
Taxes	/taxes					
Users	/users					
Client types	/client-types					

Access and Roles

K2 Replicator

1. Schema parameters Set mapping rules

2. XML entities Map data fields

3. Field mapping Map to K2 tables

4. Readiness check Validate before run

5. Start Go to import

Start Import Import settings Export settings Validate schema Back to mapping

2. XML entities

Tree view of XML structure, select nodes for mapping. Use "Open mapping" to configure fields.

Search in XML structure...

All (164) Mapped (101) Unmapped (63) Ready (53) Created (59) Manual mapping (11) Registers (5) Other (70)

With errors (164) New in XML (164) Not in schema (0) Empty (164)

<CatalogObject.BankAccount>
table / CatalogObject.BankAccount | mapped 10/15 | children: 15

<CatalogObject.Bank>
table / CatalogObject.Bank | mapped 7/12 | children: 12

K2 Replicator



Benefit:

End-to-end validation of technologies in real business scenarios.

Warehouse logistics and WMS

Warehouse documents, inventory, goods movement, employee operations and mobile workplaces. An example of a real-world solution built on K2 Platform.



Receiving

Acceptance and goods receipt



Location management

Racks, zones, storage areas



Picking

Order picking and kitting



Movement

Internal goods transfers



Inventory

Stock control and write-offs



Reporting

Warehouse analytics and current status



- Home
- CRM
- Sales
- Purchases
- Warehouse**
 - Documents
 - Inventory
 - Goods movement
 - Locations
 - Items
 - Orders
 - Reports
 - Analytics
 - Settings

Warehouse · Documents

[+ Create](#) [Post](#) [Copy](#) [Print](#) [More](#)

☐	Date	Number	Document type	Warehouse	Counterparty	Status
☐	14.10.2026	WH-000123	Goods receipt	Main warehouse	LLC Alpha	Posted
☐	13.10.2026	WH-000122	Transfer	Warehouse 2	LLC Beta	In progress
☐	12.10.2026	WH-000121	Order picking	Main warehouse	LLC Gamma	Draft
☐	11.10.2026	WH-000120	Internal movement	Warehouse 1	LLC Alpha	Posted
☐	10.10.2026	WH-000119	Inventory count	Main warehouse	—	Completed
☐	10.10.2026	WH-000118	Write-off	Warehouse 3	LLC Delta	Posted
☐	09.10.2026	WH-000117	Return from customer	Main warehouse	LLC Beta	In progress
☐	08.10.2026	WH-000116	Transfer between loca...	Warehouse 2	LLC Alpha	Posted

1 2 3 4 5 ... 248 50 Rows per page

Goods Receipt WH-000125 from 14.10.2026

General **Items (12)** Additional Comments

Number: WH-000125

Date: 14.10.2026 10:24

Warehouse: Main warehouse

Counterparty: LLC Alpha

#	Item	SKU	Quantity	Unit
1	Pump	101-001	10	pcs
2	Filter	101-002	25	pcs
3	Valve	101-003	15	pcs
4	Sensor	101-004	8	pcs
5	Seal kit	101-005	12	pcs

Post

Save

Close



Warehouse documents

Receipt, transfer, write-off, stocktake.



Real-time inventory

Control over stock levels across locations and services.



Mobile workplaces

Work with documents on mobile devices.



Storage location management

Warehouse, zones, racks, rules for placement.



Employee operations

Task execution and control.

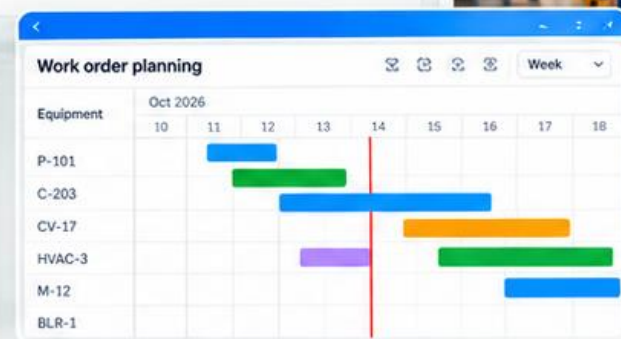
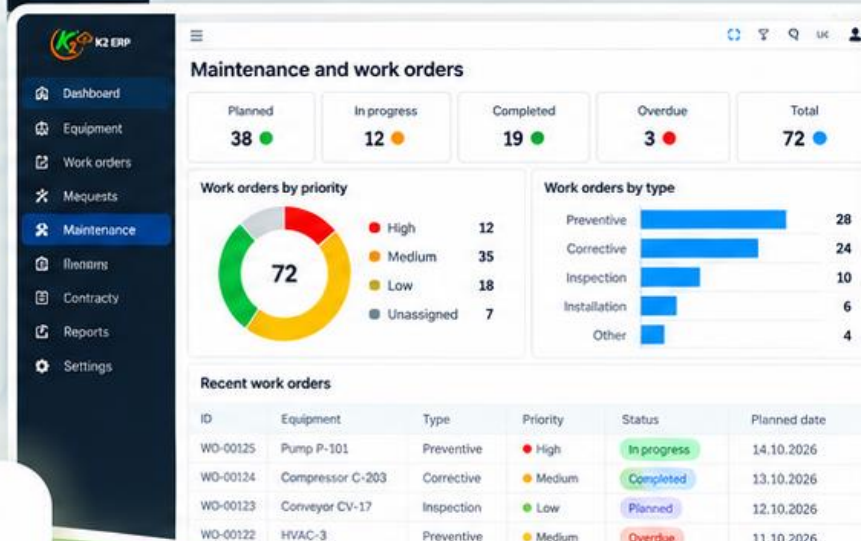
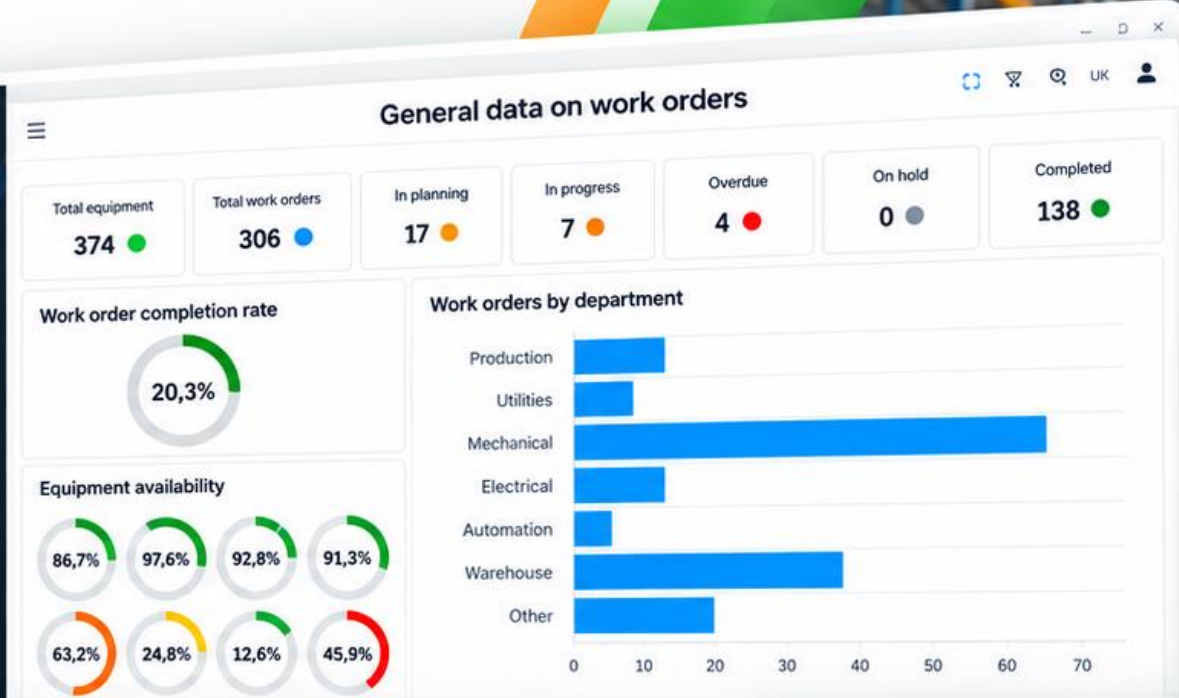
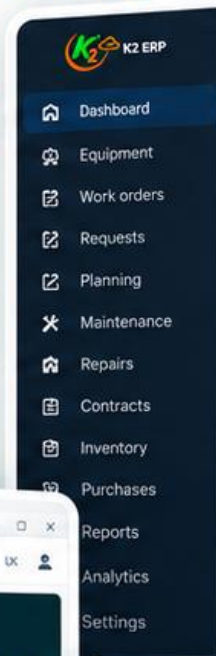
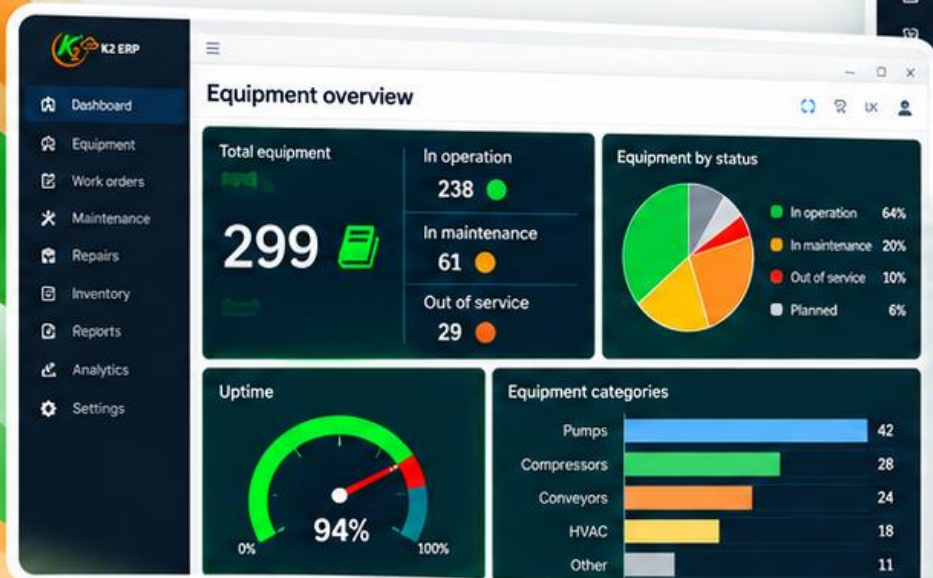


Reporting

Current warehouse status and goods flow analytics.

TOIR and equipment management

Equipment, requests, work orders, contractors, repairs, maintenance history. An example of a comprehensive application with data, documents, processes and reports.

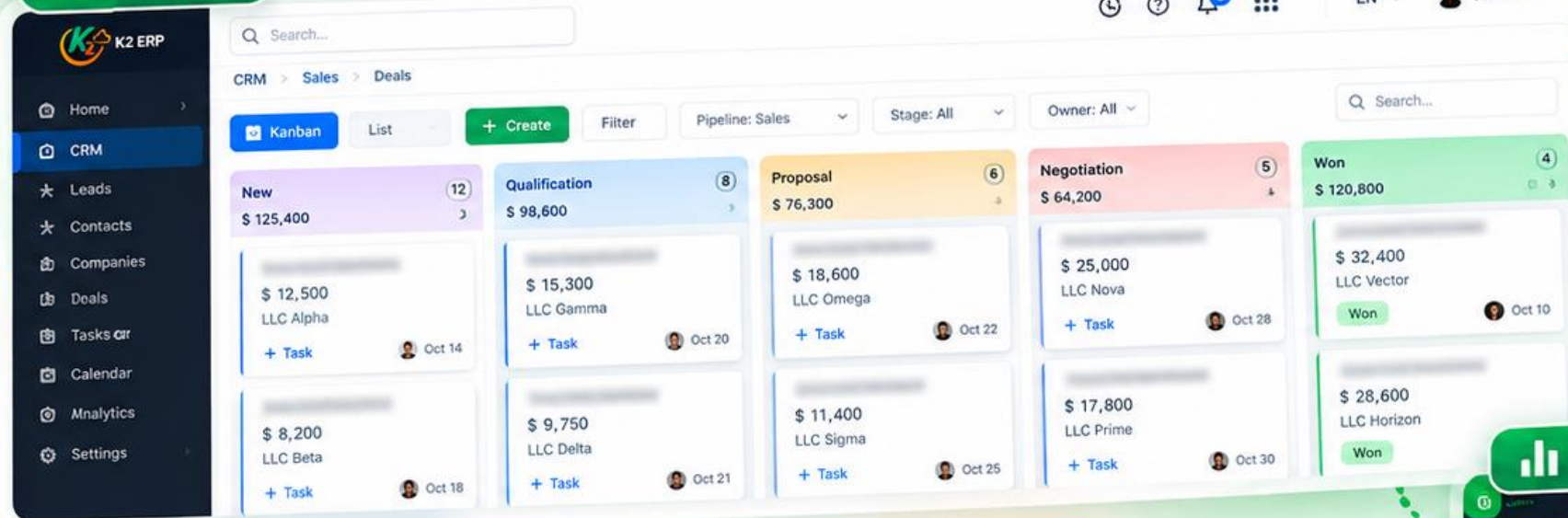


Benefit:

A single platform for complex warehouse logistics and maintenance management.

A technological platform for various business solutions: CRM, LMS, CM, LMS, CMS, e-commerce, corporate services and analytics.

CRM



Lead Card

Lead #LD-001249

Save | Contacts | History | Files

Lead name: Equipment supply

Source: Website | Source: In progress

Company: LLC Industrial | Priority: Medium

Contact person: Serhii Petrenko

Phone: +380 67 123 45 67 | Email: sp@industrial.ua

Country: Ukraine | Client type: B2B

Assigned to: Olena Kovalenko

Documentation + API

API documentation for developers

Integration and development of custom solutions using K2 platform API.

Python | JavaScript | PHP | C# | cURL | Java

```
# Get list of documents
response = client.get("/api/documents",
    params={"limit": 50})
for doc in response.json()["items"]:
    print(doc["name"])
```

Copy

K2Editor

Format | B | I | U | List | Link | Image

Equipment maintenance procedure

1. Check the equipment condition.
2. Perform diagnostic tests.
3. Record the results in the system.
4. Create a work order if necessary.
5. Attach photos and documents.



Dashboard



Pivot Analytics

Pivot table

Warehouse	Received	Issued	Balance
Main warehouse	12,450	8,320	4,130
Warehouse 2	8,210	6,500	1,710
Warehouse 3	5,980	2,410	3,570
Total	26,640	17,230	9,410



Benefit:

Reusing the platform for different business solutions.

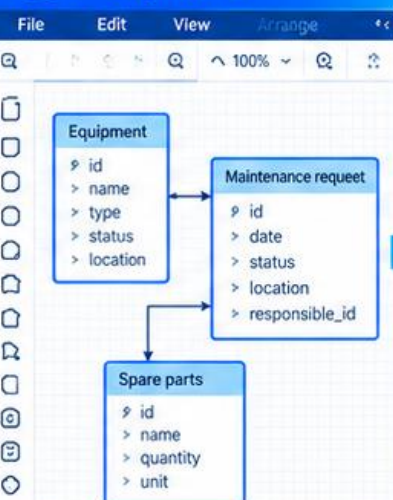


Live demonstration: creating a new application

Equipment accounting and maintenance requests — from data model to a ready web interface.
All stages are performed using K2 platform tools.



01 Data model ER / YAML



02 ORM – model generation

```
models.py x
from k2.orm import Model, fields

class Equipment(Model):
    id = fields.Integer(primary_key=True)
    name = fields.String()
    type = fields.String()
    status = fields.String()
    location = fields.String()

class MaintenanceRequest(Model):
    id = fields.Integer(primary_key=True)
    equipment_id = fields.Integer()
    date = fields.Date()
    description = fields.Text()
    status = fields.String()
    responsible_id = fields.Integer()
```

03 Database PostgreSQL

PostgreSQL 16

- Databases (4)
 - k2
 - Schemas (12)
 - public
 - Tables (6)
 - equipment
 - maintenance_request
 - spare_parts
 - employees
 - locations
 - documents
 - Views
 - Functions
 - Procedures

id	name	type
1	Pump	Pump
2	Filter	Filter
3	Valve	Valve
4	Pressure sensor	Sensor
5	Repair kit	Kit
6	Motor	Motor

04 Interface components

Components

Basic

Forms

Tables

Charts

Navigation

Additional

Search...

Button

Table

Form

Chart

Graph

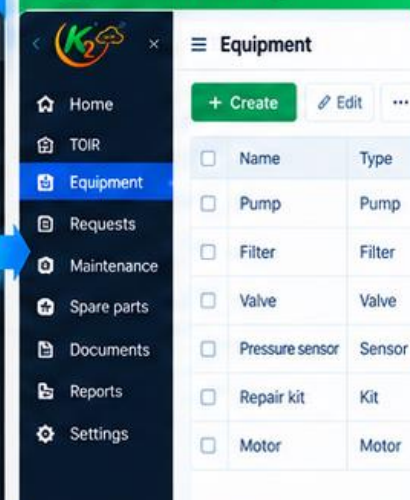
Tabs

05 Python – business logic

```
routes.py x
from k2 import route, db

@route('/api/maintenance', methods=['POST'])
def create_request(data):
    req = db.table('maintenance_request')
    .insert({
        'equipment_id': data['equipment_id'],
        'date': data['date'],
        'description': data['description'],
        'status': 'new',
        'responsible_id': data['user_id']
    })
    return {
        'id': req.id,
        'status': 'ok'
    }
```

06 Web interface and testing




Design the data structure
ER diagram in K2, export to YAML.



Generate ORM models
Automatic model generation from YAML.



Deploy the database
Tables, relationships and initial data are created.



Assemble the interface
Ready-made K2 components for fast UI creation.



Add business logic
Python scripts, routes, data validation.



Test the application
Working web interface with real data.



Fast start
From model to working solution in a short time.



Complete cycle
Data, logic and interface in one platform.



Flexibility
Easy adaptation to business requirements.



Proven in practice
Real results with your own data.

Why technological teams choose the new K2 foundation

A modern platform for building, scaling and supporting your solutions.



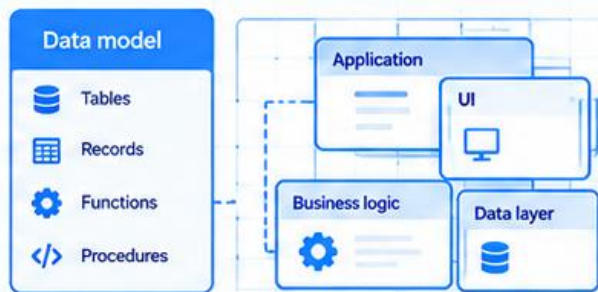
1 Replace technology obsolescence

Outdated solutions, legacy applications, large-scale customization and dependencies.



✓ K2 architecture

One platform, a single data model, separate architecture layers.



2 Replace type-specific mechanisms

Instead of a large number of modules and tools, we have a unified platform: templates, services, APIs, and events.



3 Replace outdated customized code

From a custom platform and complex modules to a unified platform with standard components.



✓ Python and TypeScript in K2

Modern developer tools, a powerful frontend and backend integration.



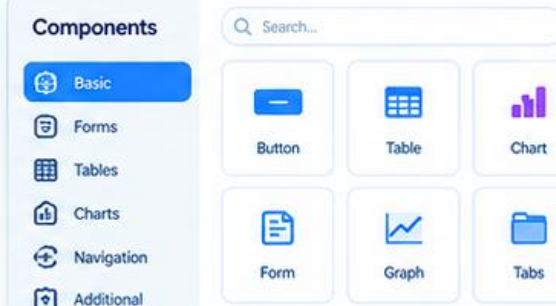
4 Single modern client

A single, accessible interface, with all the functionality and flexibility for your team.



✓ K2 components

Ready-made components, tools, visualization services and documentation in one platform.



✓ New web interface

Modern, responsive and adaptive, accessible from any device.



Benefit:



Faster implementation cycle
From days to hours.



Complete platform
Data, logic and interface in one platform.



Flexibility
Quick adaptation to business needs.



Proven in practice
Real results with your own data.

Legacy applications

Preserve experience
and business logic



Clipper
FoxPro



Delphi



1C / BAS



Access



Excel



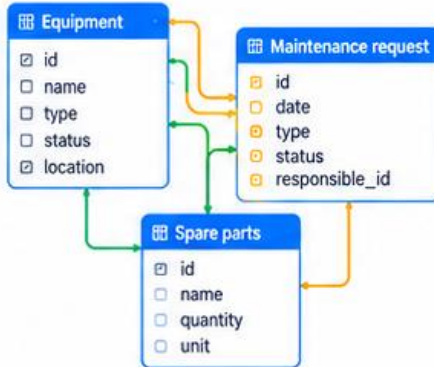
Other
systems

Analysis and modeling

Reverse engineering
and data model



ER model



Business process model (BPM)



Modern web applications

Build a new generation
of your applications



K2 Platform

Preserve business logic.
Upgrade technologies.
Build the next generation
of your applications.



IT audit

Analysis of existing
systems



Reverse engineering

Restoration of
business logic



Development

New models
and components



Integration

Connection to
existing services



Modernization

Modern web
applications on K2



Python



TypeScript



PostgreSQL

erp.kyiv.ua



K2 Platform — build your tomorrow **today!**

Ukrainian technology platform for developing, modernizing, and growing business applications.



Try K2

Get free access to the platform.

cloud.corp2.eu



Materials and demos

Articles, presentations and videos.

erp.kyiv.ua



K2 Wiki

Technical documentation of the platform.

wiki.erp.kyiv.ua



Official website

Information about the company and products.

corp2.eu

Need a dedicated cloud for your company or a custom software product on K2?



WhatsApp: **+38 067 901 63 22**



Email: **rs@corp2.net**



Developed in Ukraine. Created for the World.