Deep.Foundation

World will never be the same again because it won't have to repeat itself

What is it?

Associative graph data store. Authorization and authentication. Graph selectors based permission system. Versioned cross compatible models. Autoscaling architecture.
GUI for work with it.
Open Source.

Open ADDD API

(Associative Data Driven Development)

Associative graph as core data solution. The behaviors and handlers API is not directly accessible, everything is controlled by the state of the links.

Models

in associative graph

Versioned cross compatible models packages.

Model contains:

Collection of types for links.

Permissions for model types operations.

Handlers written on any lang code.

Model handlers

Language independent handlers of based on model data changes events. Write in GUI or your IDE.

Model based permissions

Rules written on built-in models links. Describes how links of model types can be inserted relative to each other and other models.
You can add new rules around atomic models without changing existing ones.
No need to support each possible query by permissions in advance.

Model as Package

Install any model from package registries as npm.

deepcase install deep-notify deep-notify-sms deep-notify-sms-providerName

Publish a next version of your model directly from GUI.

Autoscaling

Kuber out of box DB, GQL and handlers runtimes scaling and monitoring.

Crossplatform

One button wrap any app for
IOS/Android/MacOS/Windows/Linux
or as independent OS.
One button publish to stores.

Tree based permissions

No more roles embedded in the code! Who, on the basis of what, to what data, for what operations, in what range - All of this can be specified using graph selector rules.

What if?

Model permissions editor

Easy to create, compare, control access and compatables, handle events and write UI for models.

Model handlers editor

Write in GUI handler on any lang, and apply to events for links based on selector.

Model handler executor

Any handlers automatically up/down needed runtime processes in kuber. You need just do what you do.

Graph data customizable editor

Write queries and parses, send to views just in GUI. Wrap it and pack data layers to npm package, distribute and update from GUI already.

npm models packer

In database only random generated names for ids and tables. Only package that built this data knows business concepts. Dependent packages can deploy any business logic just after install.

What then?

No need to think about:
data architecture;
mutations and data to code refactoring;
in-project concept dependencies.

Just do what you do.

-70% of refactoring

As if you could automate the universe without rewriting

~~Code reuse~~

Before: Only programmers need it. Development has become more pleasant, but for business it has not become more flexible and cheaper.

Models reuse

After: Business needs. Allows you to isolate software solutions on the server, while maintaining the integrity and coherence of their common data.

Continuous variability

You no longer need to:
restart the project as it has reached the ceiling of its variability;
hire an architect who understands how the whole project works.

Not limited by compatibility

Any concept, its models, behavior and interfaces can be easily installed or removed from npm. Everything else will continue to work. You can always see how structures intersect with each other.

Market size

More 20 000 000 of coders
Each 1/4 new IT projects on earth in 2022

What cost?

500 000 $

12 month to complete stage 1

open and transparent development

Fully OpenSource
precedent of project
development where
humanity is a direct investor
and all parts of the project are
transparent
(yes, no any hidden part)

Stage 1

one year

PostgreSQL: partitioning, indexes, auto balanced materialized-path, nested-sets and some others
HasuraGQL
Kuber: configs and one command install/start/stop
GUI: viewer, models editor, code editor based on web vsc, packer to app or server, installer/remover/comparer
Documentation
Examples
Ready partnerships with hosting around the world for one click installation planetary solutions
OpenSource.

Stage 2

next five years

Core associative links database written on C
with incredible performance and many new features as:
Store strings, numbers, and any streams in a recognizable and searchable state. The ability to find links that are similar in structure, mentions or indirectly derived meanings. Custom analytical tools. Formation of models based on data flow.
All without custom tables or storages.
Local semantic repositories.
Fully stage 1 compatible.
Fully OpenSource.

What we want?

As **humanity**, we have a main enemy: **we cannot act together**.
Our **main goal** is to make available to humanity an **infinite flexibility and compatibility of concepts in development**. Stage 1 - at the project level.
Stage 2 - at the level of meanings.

What will?

Deep.Foundation

We will create a culture of open development and livelihood for developers around the world who have dedicated their lives to automating humanity.

Deep.Links

Solutions for work with associative data and associative models as packages in multiple package management systems.

LinksPlatform

Solutions to store associative links graph superfast, super effective.

Deep.View

Solutions for easy build react/angular/vue and others grids with inline IDE and integrated DeepGraph solutions.

Deep.Case

All solutions in one easy case.

Deep.Space

SaaS product. Cloud solution of Deep.Case to speed up the launch of projects