Before P4P Goes Live

An open-protocol proof that restaurant discovery and direct ordering do not have to be owned by the same actor that sells services on top.

2026.04.28 16:24 DENNIS HEDEGREEN BUILD OPEN-NOTE
HTTPS://HEDEGREENRESEARCH.COM/ARTICLES/BEFORE-P4P-GOES-LIVE/

Before I call this live, I want the boundary written down.

Pizza4People is not a new delivery company. It is not a finished product. It is not a replacement for Just Eat today, and it is not trying to pretend that a prototype can carry the operational weight of live restaurants, payments, refunds, delivery, support, identity verification, and customer disputes.

That matters because the temptation, especially in the week where Just Eat is leaving Denmark, is to make the story bigger than the proof.

The proof is smaller.

That is why it matters.

P4P is an open-protocol proof for restaurant ordering. The basic idea is that a restaurant should be able to announce itself publicly, and a customer should be able to find it without the first connection being owned by a closed marketplace. Discovery can be public. The menu can stay with the restaurant node. The order can go directly to that node. The registry can help a customer find the door without becoming the restaurant.

That is the line I care about.

The registry helps you find the door.

It does not become the restaurant.

What The First Proof Does

The first version proves a narrow loop.

A node announces itself to a registry. A client discovers the node through that registry. The client fetches the menu directly from the node. The client sends a test order directly to the node. If the primary registry disappears, the client can still discover through a backup registry, because the node has announced itself to both.

That is not a full business.

It is a base layer.

The distinction matters. If the registry held the menu, received the order, owned the customer account, controlled the payment, and decided which modules a restaurant could use, then P4P would just be another platform with different branding. The point is to avoid that collapse at the protocol level. The registry is there for discovery. The node is where the restaurant-facing state lives.

That means the current proof is deliberately plain.

There is a client. There are registries. There is a demo restaurant node. There is a menu endpoint. There is a test order endpoint. There are signed node announcements, so the identity direction is not just a text promise. There is an identity-log direction, so registries can eventually provide auditability without holding node private keys. There is a module-catalog direction, so future payment, printing, delivery, notification, trust, and hosting functions can exist without turning P4P into a central app store.

But the working claim is still narrow:

basic discovery and direct test ordering can be separated from platform ownership.

That is enough for a release proof.

It is not enough for live restaurants.

Why The Timing Matters

Just Eat Takeaway.com has said its operations in Denmark will conclude on 30 April 2026. That date is useful context because it makes a structural dependency visible. When a marketplace leaves, everyone suddenly has to ask where the connection actually lived. Was it with the restaurant? Was it with the customer? Or was the most basic relationship sitting inside a platform that could change terms, consolidate, sell, exit, or disappear?

The obvious question is which platform replaces it.

I think the better question is why the base layer is not open.

Restaurants can still use platforms. They can still pay for delivery, support, payment processing, analytics, loyalty systems, hosting, promotion, printing, and integrations. Commercial services are not the enemy of this idea. The enemy is the assumption that one company must own discovery, identity, menu access, and the first customer contact before any of those services can exist.

That is the platform mistake.

It turns the connection itself into rented ground.

P4P argues for a different split. Keep the base contact layer open. Let services compete above it. Let a restaurant choose modules. Let registries help with discovery. Let certification and trust claims become inspectable, signed, and replaceable over time, rather than being hidden inside one marketplace account.

That is not anti-business.

It is anti-capture.

What Is Missing On Purpose

The easiest way to oversell a prototype is to hide the absences.

So here they are.

P4P does not yet have production security. It does not have live restaurant approval. It does not have payment. It does not have delivery. It does not have tax handling. It does not have a dispute process. It does not certify that a node is a real restaurant. It does not certify module providers. It does not solve support. It does not make a public demo node into a production restaurant system.

Those are not footnotes.

They are the release boundary.

A real restaurant system needs an operator layer where a restaurant can choose whether it is open, whether it accepts orders, which modules are active, what key material signs its announcements, and what backups or hosted services it delegates to. A real trust layer needs more than a public key. It needs claims that can be reviewed, revoked, and audited. A real module ecosystem needs provider manifests and data-access declarations, so a printer module, payment module, delivery module, or business verification module cannot quietly become the owner of discovery.

Those are next layers.

They should not be smuggled into version 0.1 by language.

Version 0.1 is the proof that the connection can be split correctly.

What The Public Proof Needs To Show

The public proof is simple enough to be judged.

One public client.

Two public registries.

One signed demo node.

The client discovers the node through the primary registry. The client fetches the menu directly from the node. The client sends a test order directly to the node. The primary registry is stopped or blocked. Discovery still works through the backup registry.

That is the proof.

Not a pitch deck.

Not a marketplace.

Not a promise that restaurants can use it tomorrow morning.

A working demonstration that the basic contact layer does not have to be owned by the same actor that sells the services on top.

If that proof is clean, then the next conversation becomes useful. Journalists can ask what this means for Denmark after Just Eat. Technical people can ask whether the protocol boundaries are sane. Restaurants can ask what operator control would look like. Module providers can ask where payment, printing, delivery, POS, notifications, business verification, and hosted backup nodes fit. Platforms can ask whether they want to compete as service providers on an open layer instead of owning the whole silo.

Those are the right questions.

But they only stay right if the first release stays honest.

The Release Claim

So the claim before release is this:

P4P is a small working proof that restaurant discovery and direct ordering can be organized as open infrastructure instead of as a closed marketplace dependency.

It does not solve the whole restaurant economy.

It does not need to.

The first job is to prove that the base relationship can be held differently: registry for discovery, node for restaurant state, direct menu, direct test order, node-owned keys, and a path toward modules and trust that does not require one central company to own the relationship.

That is the release I am willing to stand behind.

Everything else has to earn its place after that.

— Dennis Hedegreen

RELATION MEMORY

MATURES FROM Download Was the Wrong Door · P4P applies the corrected public-door principle to a protocol-style tool rather than a static app download.

NEARBY A Tool Can Be Right and Wrong at the Same Time · P4P keeps the same tool-boundary discipline: useful public proof does not equal full market solution.

CANONICAL URL: HTTPS://HEDEGREENRESEARCH.COM/ARTICLES/BEFORE-P4P-GOES-LIVE/
PDF VIEW GENERATED: 2026-07-11T15:35:45.735Z