Nosh Gateway Provider Infrastructure

LEO STEPANEWK, Princeton University, USA
STEPHEN DONG*, Princeton University, USA

We detail our design and implementation of the Gateway Provider (GP) for Nosh's distributed food-ordering network. Nosh is a startup that aims to decentralize food ordering away from platforms like DoorDash. They are designing an open protocol where customers and restaurants can engage with each other without a rent-seeking intermediary. The GP indexes the independent servers operated by restaurant co-ops and provides a simple API for entrepreneurs to participate in the network. We explore the advantages and limitations of the GP architecture and alternative design choices. Despite being a point of centralization, we argue that our GP design does not enable a monopoly because key restaurant data is stored in a decentralized smart contract that is public and independent of Nosh. We pose several questions and next steps surrounding the GP for Nosh to consider as they continue to develop the network.

ACM Reference Format:
Leo Stepanewk and Stephen Dong. 2024. Nosh Gateway Provider Infrastructure. 1, 1 (May 2024), 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Delivery services [i.e. DoorDash, UberEats] have an incentive towards profit, and they have successfully exploited restaurants with high commissions and fees. In 2020, if you ordered through one of these platforms, you were likely assessing a surcharge up to 35% [11], significantly eroding the already slim profit margin of the order. Additionally, there is a breadth of recent work uncovering the exploitation of gig workers by these centralized companies. [14]

Despite these problems, entrepreneurs have a difficult time disrupting this system because DoorDash and other companies have lock-in advantages over restaurants. Starting a competitor from scratch requires one to onboard hundreds of restaurants before the application becomes useful, and still, these incumbents will have more options for customers. Nosh builds upon a trend of restaurants, drivers, and customers who desire a better experience.

Nosh operates as a food-ordering service that works with local independent restaurants and promotes non-exploitative business practices within the industry [1]. To do this at scale, Nosh is building an open protocol for decentralized commerce in the food industry. This protocol would allow customers and restaurants to make transactions without an intermediary (thus without the exorbitant service fees). This framework will be extensible to online commerce in general.

To support Nosh's vision as it continues to scale, we propose and implement a Gateway Provider microservice [9], which abstracts redundant logic in dealing with data from provider logic to allow developers to focus on developing core functionality. We believe that this service will enable Nosh to be more scalable and improve the public developer experience.

*All authors contributed equally to this research.

Authors' addresses: Leo Stepanewk, Princeton University, Princeton, NJ, USA; Stephen Dong, Princeton University, Princeton, NJ, USA.

Manuscript submitted to ACM

This paper serves a dual purpose: (1) to justify and explain the design decisions underlying the Gateway Provider architecture that we created and (2) to provide an overview of Nosh's system that will help a potential new collaborator of Nosh quickly get acquainted with the high-level technical overview of the network.

2 BACKGROUND

2.1 Peer-to-Peer Network Design

Most services on the web today operate through a centralized network where requests, data storage, and processing are performed by a single entity (such as a company). While this architecture offers advantages like manageability and efficiency, it comes with significant downsides for the end user including censorship and monopolization of services.

Peer-to-peer (P2P) networks solve the issues with centralization by delegating storage, computation, and request handling to a large set of publicly owned servers. The nodes are generally identical, with no one entity having key control of the network, and may have high failure rates. However, P2P architectures introduce technical challenges such as how to route traffic, resist node failures, and index all the nodes. For instance, Chord [15] demonstrates a P2P data storage architecture that allows users to retrieve files stored across a fully decentralized network of nodes.

2.2 Blockchain and Smart Contracts

Blockchain technology builds on the concept of P2P networks by providing a shared, immutable ledger that enables tamper-proof data storage in a decentralized system. The Bitcoin whitepaper [13] introduced a blockchain-based, P2P protocol for electronic money that did not rely on a centralized processor and maintained reliability in an environment where users are incentivized to tamper with transactions. Since then, the blockchain and cryptocurrency ecosystem has created extensions to general-purpose data storage and computation. The Ethereum project [12] popularized the notion of a smart contract, a computer program with persistent data storage that exists on the blockchain. Smart contracts store their variable contents and source code publicly, meaning that anyone can read the data stored inside and inspect its inner workings. The smart contract's functions can be invoked by any user (unless code is written to give only permissioned access) and the computation is executed and verified in a decentralized manner.

2.3 API Providers for Decentralized Systems

To interact directly with a blockchain like Ethereum, users need to host a node that adheres to the network specification. API providers have emerged to offer convenient access to blockchain networks by managing the node infrastructure themselves. Infura [5] is a popular Ethereum API provider that gives developers a simple interface for querying data from the network with a traffic-based pricing model [6]. Similar to Infura, Nosh is building a decentralized protocol in which users (both local entrepreneurs trying to build an application, and restaurants trying to form a cooperative) can host nodes that adhere to this specification and are connected to the blockchain network.

2.4 Open Source Protocols

We have seen a host of open-source protocols applied to different contexts where we have witnessed the early success of the open-source system. For instance, BlueSky [2] is a decentralized microblogging social platform (akin to Twitter) with the vision of decentralizing power from a single authority (often a company). Beckn [4] is developing an open protocol for commerce with the vision of empowering sellers. Similarly, Nosh is aiming to build an open-source protocol applied to the food delivery commerce space.

3 ARCHITECTURAL OVERVIEW AND TERMINOLOGY

3.1 Terminology

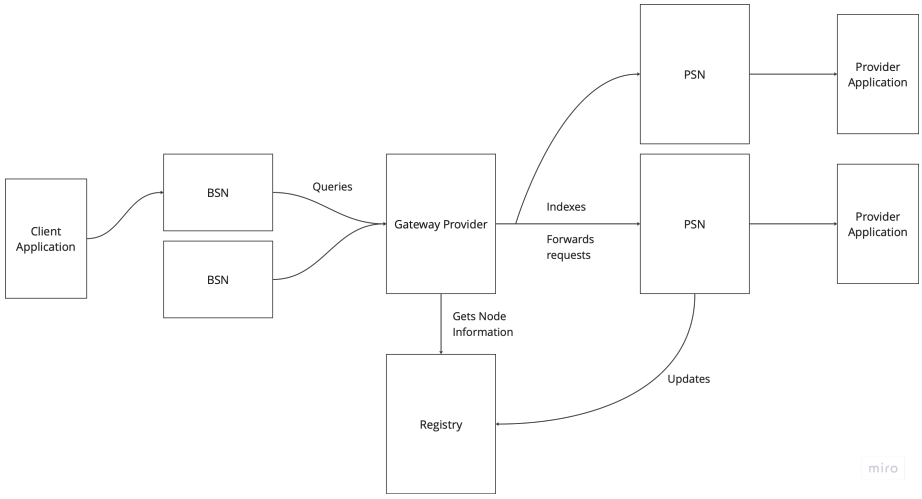

Fig. 1. **Simplified graphic of Nosh's network architecture.**

Within the graphic, there are 4 components: the **Buyer Service Node (BSN)**, the **Provider Service Node (PSN)**, the **Registry**, and the **Gateway Provider**. All components are free and open-source. Below, we define these terms as well as additional terminology that would be useful in understanding the Nosh network.

3.1.1 Buyer. A buyer is an individual who wishes to purchase a product or service. For instance, they could be a user who wants to order food from a restaurant.

3.1.2 Provider. A provider (also referred to as a seller) is an organization that has a catalog of products or services available for sale (i.e. a restaurant).

3.1.3 Entrepreneur. We refer to a person building a food-ordering app on the Nosh network as an Entrepreneur.

3.1.4 Buyer Supporting Node (BSN). BSNs are servers that store buyer data (such as personal information and saved menu items) within the network. These would be hosted by entrepreneurs as part of their application infrastructure.

3.1.5 Provider Supporting Node (PSN). PSNs are servers operated by providers for the seller side of the network. They respond to queries about the catalog of items for sale and process operations like placing an order.

3.1.6 Polygon. Each PSN stores a polygon, which is a 2-dimensional planar region that defines the region by which a PSN serves (see Figure 2).

3.1.7 Decentralized Registry. The registry is a decentralized public ledger that maintains the list of all provider nodes with their URLs, service area polygons, and other metadata. It is implemented as a Solidity smart contract.

3.1.8 Gateway Provider (we are building). The Gateway Provider (GP) offers a simple API interface to enable the application to forward requests to all relevant PSNs. It is a server that maintains a local PSN database and synchronizes with the registry.

3.2 Network Overview

In this section, we walk through the flow of Nosh's architecture (Figure 1). The purpose of this section is for a high-level understanding of how these components interact with each other. For a more detailed/technical description, please refer to Nosh's transactions lifecycle RFC [7].

To start, two primary groups are being served, the **Buyers** and the **Providers**. The **Entrepreneurs** develop new food-delivery applications to serve the Buyers and operate **BSNs** to store user data and talk to the network. On the other hand, the Providers host their data on **PSNs** to indicate their presence and available items for sale in the network. These PSNs are operated by small business cooperatives or local chambers of commerce.

A Buyer communicates their intent to a BSN through the Client Application. The Buyer and BSN might execute some request, i.e. search for local "Pizzas". This search will form a query alongside metadata like the user's location. The query is sent by the BSN to the **Gateway Provider** (GP), which forwards the request to all relevant PSNs. Each PSN will then send a list of Providers (i.e. restaurants) back to the GP. After the GP has received the relevant Providers from each PSN, it will aggregate the data to send back to the BSN and client application such that the Buyer can select the PSN and provider that they want to interact with moving forward. The remaining logic (such as placing an order) is handled directly via communication between the BSN and PSN.

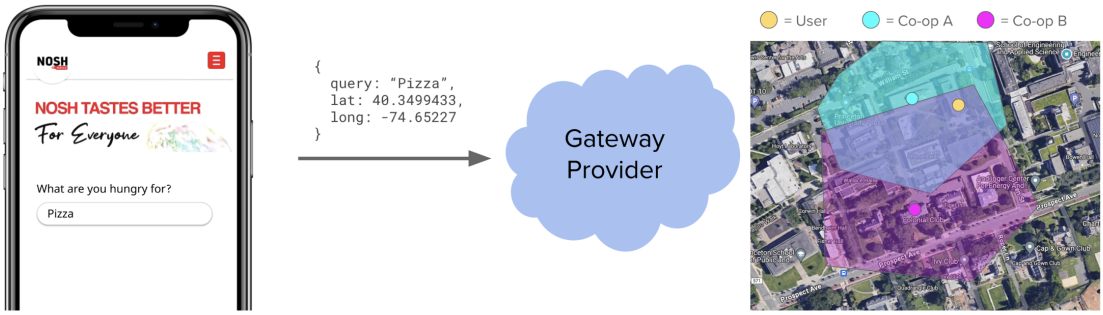

Fig. 2. The flow of a sample search query from an application to the gateway provider. The GP references the PSN polygons when deciding what results to return to the user. The user's location is in yellow, while two co-op service areas are labeled in blue and pink.

3.3 Gateway Provider (GP)

A core operation performed on the Nosh network is forwarding search requests from the entrepreneur's application to all PSNs that are nearby. However, the PSN metadata is stored in a `mapping` data structure within the smart contract, which is not efficiently queryable. To retrieve the relevant PSNs, the only option is to extract each PSN iteratively from the map and then check whether the user's current location falls within its polygon.

The GP offers an intermediate service that indexes the PSNs ahead of time and stores them within a Postgres+PostGIS [8] database for efficient spatial queries. It regularly synchronizes with the registry contents and refreshes its internal database with the new PSN metadata, including the polygons. When the GP receives a search request, which contains a search query and the user's current location, it performs a spatial query over the PSNs and selects the ones where the

location falls within their polygon. The search request is then forwarded to the selected PSNs and their responses are aggregated and sent back to the user.

Without the GP, entrepreneurs would have issues identifying and forwarding requests to the right PSNs among all the independent servers located on the network. It would take time and money to write their own software and host the infrastructure to handle requests. There will also be considerable complexity in communicating with all the different network components. The GP reduces network participation for entrepreneurs down to a single API call. We created a prototype of the GP infrastructure for Nosh, who will be hosting the first version on the network for public use.

3.4 Design Considerations

One concern with this architecture is that having a GP introduces a point of centralization. While the GP still inherits issues such as potential server downtime, we do not believe it enables the monopolization of the restaurant ecosystem the way DoorDash and other food delivery platforms currently do. The competitive advantage of the established platforms is their control over the data and ordering interfaces of restaurants. In the Nosh network, these components are stored publicly and immutably in the decentralized registry. Since this registry is completely transparent and exists independently of Nosh, no single entity has control over the information that could be monopolizable. Any individual can create a GP that performs all the necessary operations for their application. Additionally, if someone wants to interact with a single PSN on their own, they do not need a GP at all. The PSN server URL is located in the registry and an order can be placed directly.

The vision is that multiple gateway providers will provide services on the network in a competitive market that minimizes rent-seeking behavior. These different entities can make their own decisions about how to organize the PSN data from the registry. Some gateway providers might focus on specific geographic regions or cuisines, while others will index all the nodes. When building their applications, entrepreneurs can register for API access with their choice of gateway provider, the same way an Ethereum user can register with Infura.

4 IMPLEMENTATION

4.1 Registry Indexing Service

The first step of our implementation involves synchronizing the registry with a local Postgres database. We wrote a program in TypeScript that imports the interface of the registry contract and polls its list of PSNs for updates every minute (by iterating over all map keys). We insert the data from every new PSN into a table with the following schema:

```
model NodeEntry {
  uid           String      @id
  name          String
  callbackUrl   String
  industryCode  String
  owner         String
  nodeType      NodeType
  status        NodeStatus
  coords        Unsupported("geometry(Polygon, 4326)")
  @@index([coords], name: "location_idx", type: Gist)
}
```

In the registry smart contract, the PSN polygons are represented in a compressed h3 index [3] format to make them more cost-efficient to store on the blockchain. Part of the synchronization process involved converting the h3 indexes into polygons with points defined by latitude and longitude. These polygons can be inserted into the table with the PostGIS geometry type for geospatial indexing. We used Prisma as our database object mapping library and Docker to manage our PostGIS instance.

4.2 Gateway Provider Server

After creating the registry synchronization mechanism, we wrote a prototype of the actual gateway provider server. Using TypeScript and Express, we created a search API endpoint that receives a query and the user's current latitude and longitude in JSON form. It performs a geospatial query on the Postgres database to find relevant PSNs, forwards the search request to them, aggregates the results, and returns them to the application. In our design, the GP reaches out to the PSNs every time there is a search request. This minimizes the amount of state that the GP needs to store at the expense of more network messages and latency. We describe an alternative protocol that Nosh could consider in section 6.2 which does not require the gateway to reach out to PSNs on every request (with the tradeoff of needing to store more data locally).

5 LIMITATIONS

In this section, we describe two different types of limitations throughout our implementation of the Gateway Provider. **Project Limitations** describe the limitations in what we were able to accomplish in the timespan we had worked on the project, whereas **Architectural Limitations** describe limitations that have been created through our design decisions in creating the Gateway Provider.

5.1 Project Limitations

5.1.1 Adding new nodes to the registry. Right now, Nosh is the owner of the smart contract and has permission to add new PSN entries. This is a fine solution for the initial growth of the network, but as the protocol scales up with more users there needs to be a decentralized approval mechanism for new nodes. Nosh cannot be a point of centralization and will not feasibly handle all node approvals themselves. One solution is to establish a governing body like a decentralized autonomous organization (DAO) that works on regulating PSNs in a transparent and distributed way.

5.1.2 Server billing. We assume that, under the current architecture, servers can bill each other and the end user for the services provided. While this might work for a small network scale, Nosh would like to establish a token-based economic system that routes payments automatically through the blockchain.

5.1.3 Misbehaving nodes. Currently, there is no explicit solution for misbehaving nodes. Ideally, a decentralized governing organization could be responsible for addressing these issues. There are also ideas surrounding an on-chain reputation process that automatically regulates node inclusion.

5.2 Architectural Limitations

5.2.1 Traffic directed through Gateway Provider. Because each BSN has to interact with a Gateway Provider, there will be a lot of traffic directed toward the Gateway Provider Server, especially if there is only one on the network. As the maintainers of the Gateway Providers would have to deal with this traffic, this creates more costs for those using the service from either Nosh or a third-party developer.

5.2.2 Latency from request forwarding. In the current implementation, the Gateway Provider has to forward the user's request and receive replies from all relevant PSNs to respond to a given search query. The response speed becomes dependent on the speed of the slowest PSN server to reply, which could be a source of latency.

6 NEXT STEPS

6.1 Expand search endpoint to handle targeted queries

We have implemented a simple search endpoint that allows the user to broadly search for menu items in PSNs through the gateway (e.g. pizza). However, many application use cases involve searching for a specific restaurant or menu item that was previously ordered. In this case, it might be valuable to enhance the search endpoint to have the functionality to search by PSN or item ID. Adding this feature will allow more flexibility in how developers want to manage relationships between BSNs and individual PSNs.

6.2 Investigate using pub/sub messaging between PSN and GP

Our current implementation of the GP makes it such that when a client wants to search for a restaurant, the GP would have to query through all of the PSNs to aggregate the PSNs that fit the search criteria. In 5.2.2, we discussed how this creates additional latency whenever a request is made. A potential alternative would be to utilize the publish-subscribe model [10], in which PSNs periodically (or when there is a local update) push their relevant data (i.e. menu items) to the GP. This would allow the GP to reply to search requests directly instead of making repeated calls to the actual PSNs, eliminating that latency component. The trade-off of this is that the GP would be responsible for storing the data (e.g. menus) of the PSNs it has indexed. We recommend further investigation of this alternative design to potentially decrease latency.

7 ACKNOWLEDGEMENTS

We would like to thank Michael Perhats from Nosh for helping us set up our development environment, understanding Nosh's codebase to quickly ramp up this project, and providing us with an amazing educational experience. Additionally, we would like to thank Professor Andrés Monroy-Hernández, Dan Calacci, and the rest of the teaching staff as well as students of EGR 371 (Designing the Future of Work, Public Interest Technology Development) for enabling us to work on this project, providing us feedback, and helping us learn about building public-interest technology projects.

REFERENCES

[1] [n. d.]. About Nosh Delivery - Online Ordering, takeout, and restaurant delivery to Northern Colorado – noshdelivery.co. https://www.noshdelivery.co/about_us
[2] [n. d.]. Company - Bluesky — bsky.social. https://bsky.social/about/faq
[3] [n. d.]. H3. https://h3geo.org/
[4] [n. d.]. Home - Beckn protocol — becknprotocol.io. https://becknprotocol.io
[5] [n. d.]. Infura. https://www.infura.io/
[6] [n. d.]. Infura Pricing. https://www.infura.io/pricing/
[7] [n. d.]. Nosh Transactions Lifecycle RFC. https://github.com/Palette-Labs-Inc/registry/blob/main/rfcs/00001-lifecycle-apis.md
[8] [n. d.]. PostGIS — postgis.net. https://postgis.net/
[9] [n. d.]. What are Microservieces? ([n. d.]). https://aws.amazon.com/microservices/
[10] [n. d.]. What is Pub Sub? - Pub/Sub Messaging Explained. ([n. d.]). https://aws.amazon.com/what-is/pub-sub-messaging/#:~:text=The%20publish%2Dsubscribe%20model%20reduces,be%20delivered%20to%20which%20endpoints.
[11] 2020. Restaurant-owned Nosh is a different delivery service. (2020). https://boulderweekly.com/cuisine/restaurant-owned-nosh-is-a-different-delivery-service/

[12] Vitalik Buterin. 2013. Ethereum white paper: a next generation smart contract & decentralized application platform. (2013).
[13] Satoshi Nakamoto. 2009. Bitcoin: A peer-to-peer electronic cash system. (2009).
[14] Hi-Phi Nation. 2022. The Problem with Gig Work. https://www.podchaser.com/podcasts/hi-phi-nation-240453/episodes/the-problem-with-gig-work-171533064
[15] Ion Stoica, Robert Morris, David Liben-Nowell, David R Karger, M Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. 2003. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. (2003).