Labour Platform Architecture

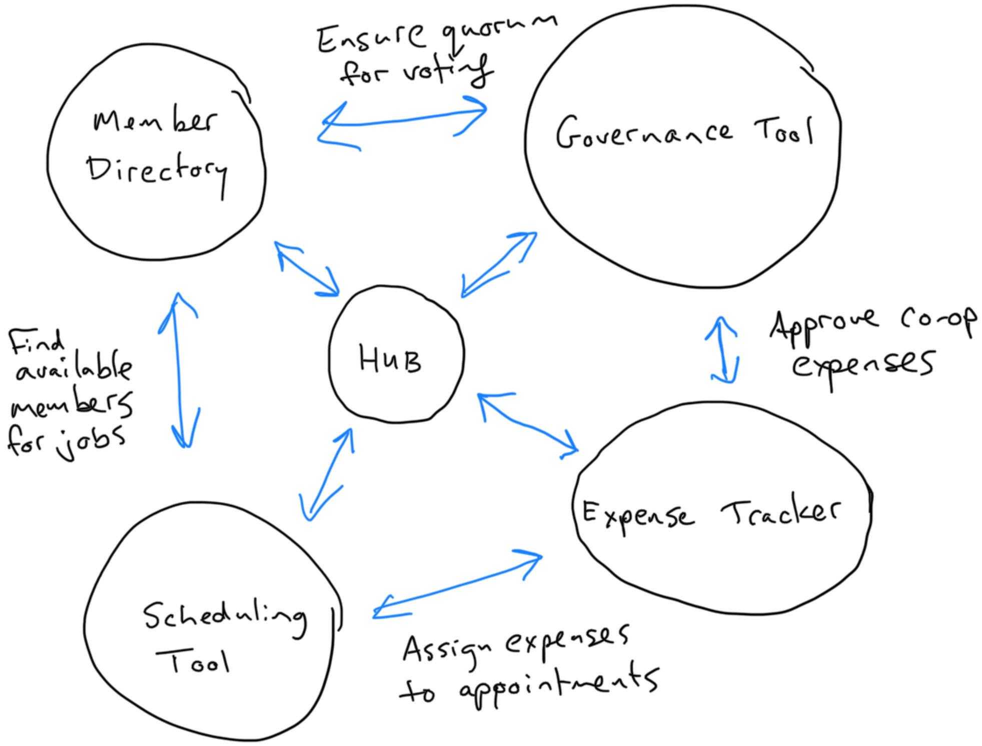
The Labour Platform is envisioned as a collection of modular web applications which address the different needs of workers in the platform co-op ecosystem, connected via a central Labour Platform Hub. These applications may include tools for governance, expense tracking, and scheduling. Where possible, we want to build upon existing open source tools and allow them to be used together through the Hub, which will provide customizable reporting on the activities of a platform cooperative for its member-owners. The Labour Platform's modules must work well together and as such must use a shared authentication system to facilitate ease of access for member-owners. They must have well-documented application programming interfaces (APIs) which facilitate inter-application communication as part of a modular system. Lastly, and most critically, they must serve a global user base, both by providing functionality that serves different languages and regions, and by performing well in environments where users may be limited by costly or intermittent internet access and slow devices.

Platform

As we hope to build upon and adapt existing tools to become part of a Labour Platform as well as developing new web applications, a requirement of this architecture is to be fairly agnostic in terms of application framework. Some tools we have begun to examine are built with Laravel (PHP), some with Expres (NodeJS), and others with Ruby on Rails. We have not yet settled on a language and framework for the new web applications we will be developing.

As the needs of platform co-ops vary widely by sector and region, we envision that different co-ops will need different modules. This is the rationale for building a modular collection of tools, as we want to enable platform co-ops to assemble a set of modules that is relevant to their needs rather than building a monolithic application that burdens co-ops and their member-owners with unnecessary functionality.

A core component of the Labour Platform will be the Labour Platform Hub. In its most elemental form, this will simply be a website where co-op member owners can go to quickly access other web applications that they need to use as part of their work. Based on feedback from our cooperative partners, a broader goal of the Hub is to provide a flexible reporting interface where data from other modules of the Labour Platform can be accessed by member owners — for example, aggregate hours worked/clients served within a reporting period, aggregate expenses logged, total member owners within the co-op, et cetera. Decisions about the kinds of data that should be made available through the reporting interface will be determined by co-design activities with partner organizations.



The Labour Platform Hub is connected via API to Labour Platform modules, and modules can also provide data directly to each other via API connections.

Success Criteria for Labour Platform Modules

As identified above, any adapted or newly-created web application should meet the following success criteria:

1. It should have a robust and well-documented API that supports the reporting needs of the co-op.
2. It should support authentication via third-party service so that a co-op can use a single authentication mechanism for its entire Labour Platform.
3. It should support internationalization, both by exposing the user interface for translation and by supporting different regional considerations (currencies and taxes, regulatory requirements, etc.)
4. It should provide a performant experience for users regardless of device or internet connection speed.

Success Criteria 1: Application Programming Interface

A robust and well-documented API will be a critical component of all Labour Platform modules. For example, an expense-tracking tool should provide an API that allows an authenticated user to access their expense data via the Labour Platform Hub. APIs support the modular nature of the Labour Platform, acting as the glue that connects the modules together and provides a cohesive experience, ensuring that relevant data is always available throughout the Labour Platform.

The Labour Platform Hub is connected via API to Labour Platform modules, and modules can also provide data directly to each other via API connections.

Success Criteria 2: Authentication System

Assembling a Labour Platform from different modules necessitates a shared authentication system. If a member-owner needs to access multiple tools, they must be able to do so with the same credentials, whether these credentials are a common username and password or authentication via a third-party account (e.g. Facebook or Google). We are interested in the potential of Auth0 as a tool for enabling this authentication system. Auth0 provides software development kits (SDKs) for most major web application programming languages and frameworks, and allows applications to be configured to support authentication via username and password or via third-party accounts. In our examination of existing tools, we are particularly interested in applications which have an authentication system that can be modified to support shared authentication with other modules in the Labour Platform.

Success Criteria 3: Internationalization

Platform co-ops are a global phenomenon, and as such we need to keep internationalization first and foremost in our development of Labour Platform modules. We intend to develop all Labour Platform modules as open source software, which will enable us to leverage the open source plans offered by translation tools such as Transifex to facilitate the translation of their user interfaces. For any modules which involve currencies, taxes, or regulatory aspects of platform co-op business, we will need to ensure that the development process supports regional customization for local currencies, tax regimes, and regulations.

Success Criteria 4: Performance

The performance of Labour Platform modules is essential to their adoption and use by co-op member-owners. We intend to develop all new modules as Progressive Web Applications (PWAs) and leverage PWA features to provide a performant experience on mobile devices with limited connectivity, including supporting as much functionality as possible in robust offline modes. This could include the ability to log an expense while the device is offline and uploading it once the connection is restored, or caching the details of an upcoming appointment on the user’s device so that they can navigate to their client’s residence even in areas with limited service.

Another strategy for performance is the use of a performance budget. We intend to deliver minimal client-side assets to achieve PWA functionality, and use aggressive asset caching to ensure that the application can be loaded with minimal requests following its first run. We will leverage tools such as Lighthouse to establish targets for performance and ensure that Labour Platform modules meet these targets.