Data dashboards with R and Shiny

David Henderson, Chief Data Officer
Family Independence Initiative
david@fii.org @david_henderson

Family Independence Initiative

Stuff you should download

https://github.com/dhenderson/shinyheinz

Make sure you have the following R packages

- shiny
- dplyr
- ggplot2
- tidy
- leaflet
- plotly

What is Shiny?

- Shiny is an **R** package that makes it **easy** to build interactive web applications (apps) straight from R.

- Shiny apps are **single-page-apps**.

- Shiny apps are built using a **reactive** programming paradigm.

Why should you care?

- Get R into the **browser**!
- Build web apps with R only (no need to know HTML, CSS, Javascript, etc.)
- Empower people to apply your analysis and **explore data on their own**.
- One-click deployment with shinyapps.io

About the Family Independence Initiative

- 1,500 low-income households submit monthly journals online

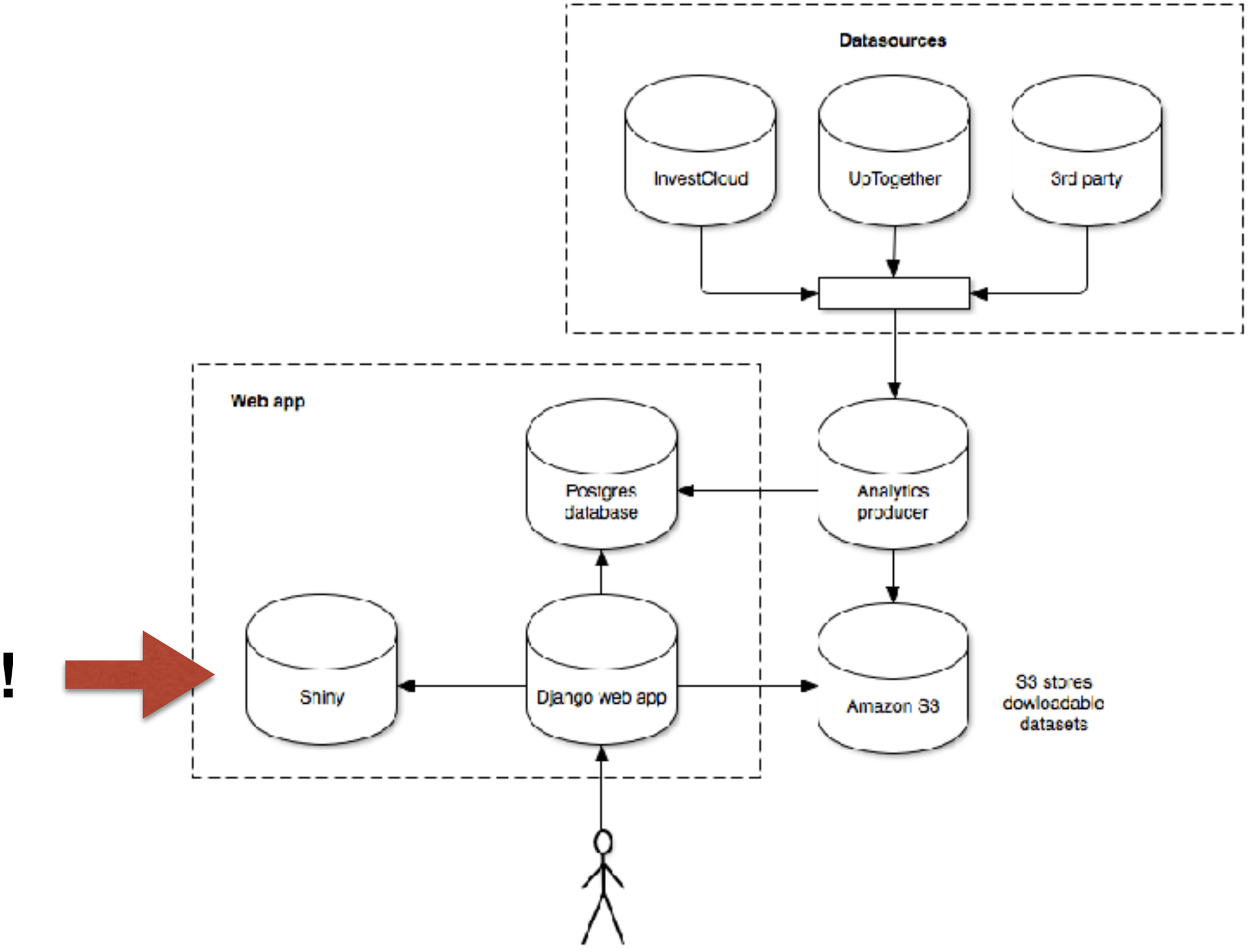
- ~300 data points in various domains

- We invest in families' initiative

Family Independence Initiative

Shiny is the last mile of our data pipeline

Shiny!



Life before Shiny

- Data requests were handled **one at a time**.
- Manually processing each request means your team is **treading water**, not focusing on higher order problems.

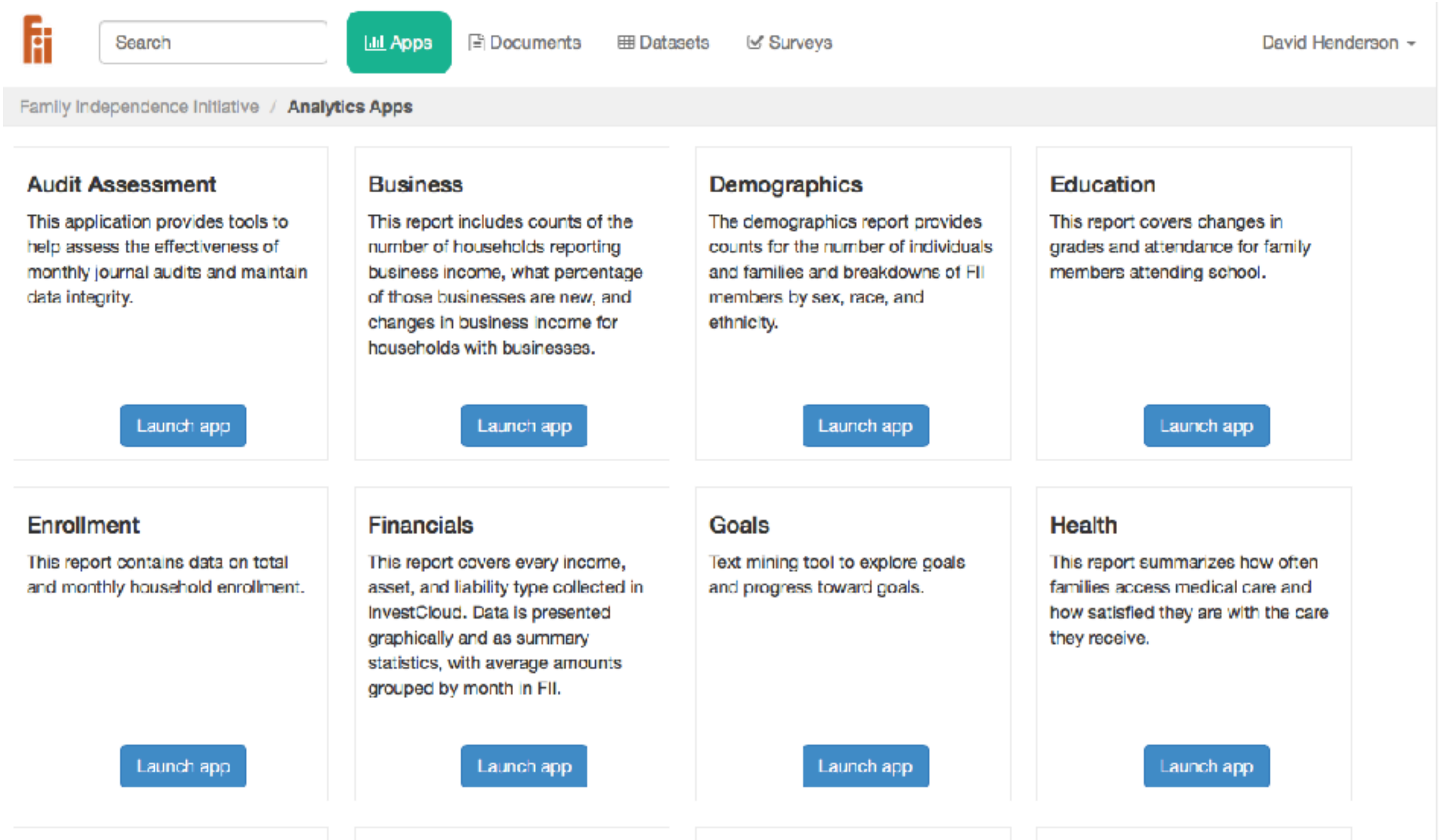
Life after Shiny

- Staff and partners **answer their own** questions through Shiny apps.

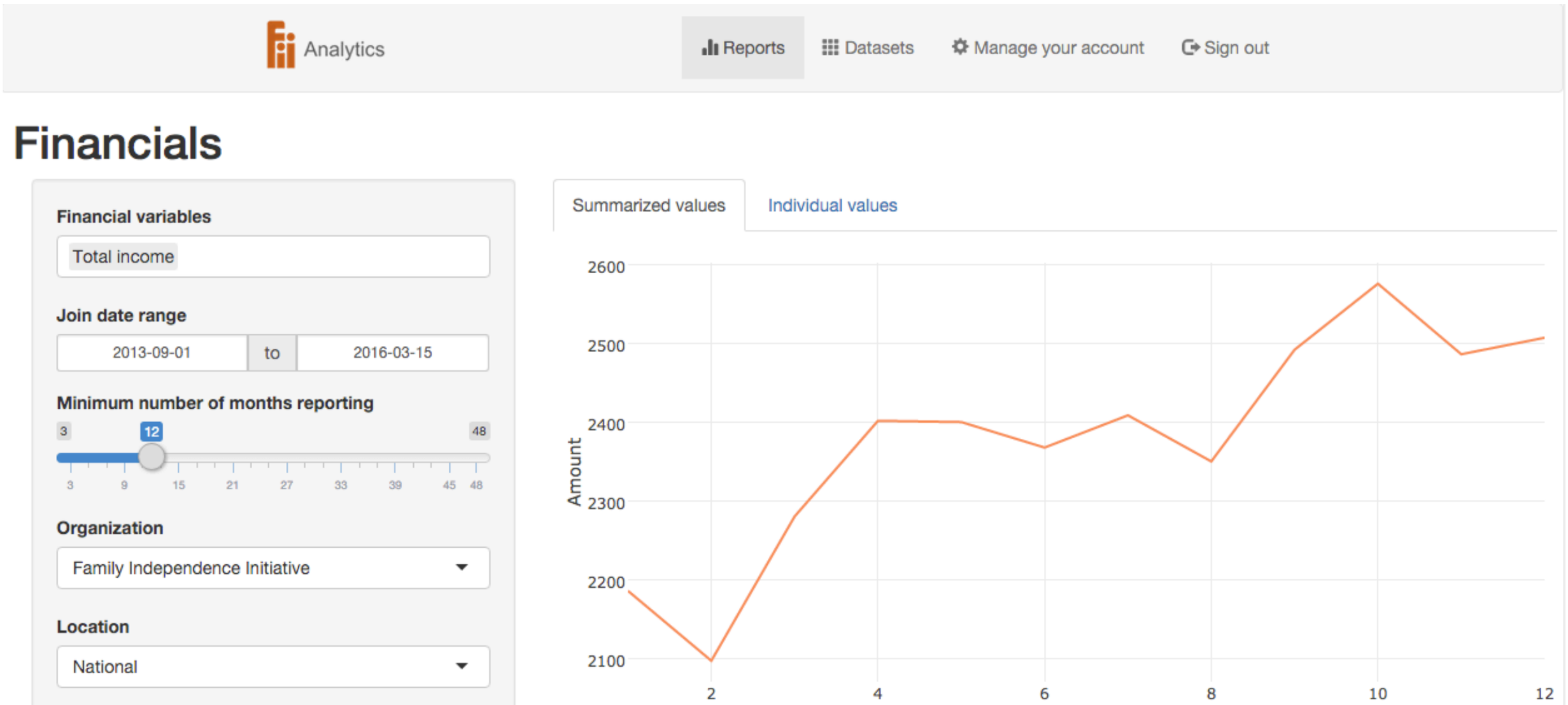
- As we answer new types of questions, we build **new Shiny apps**.

- For any type of analysis, we **write once** and **run forever**.

We have multiple Shiny apps



Each app focuses on a particular set of questions



How do you build a Shiny app?

- Download R Studio

- Install the “shiny” package

Shiny apps are made up of two R scripts

- **ui.R** - This script holds all the UI code
- **server.R** - This script holds all the “server” code
- That’s it! (although you can get fancier)

Shiny apps are reactive

- As a user changes UI input, those changes are sent back to the server without you needing to do anything and without form submissions.

Shiny apps are reactive

- As a user changes UI input, those changes are sent back to the server without you needing to do anything and without form submissions.

- Reactivity means you just need to focus on the state of your app at any point in time without worrying about the client/server plumbing.

Shiny apps are reactive

- As a user changes UI input, those changes are sent back to the server without you needing to do anything and without form submissions.

- Reactivity means you just need to focus on the state of your app at any point in time without worrying about the client/server plumbing.

- **Shiny automagically handles updating your data, calculations, and charts for you.**

Let's take a look at the **ui.R** and **server.R** files

ui.R

```
library(shiny)

shinyUI(fluidPage
  # Your UI code goes here
))
```

server.R

```
library(shiny)

shinyServer(function(input, output){
  # Your server code goes here
})
```

The server takes an anonymous function with two arguments, **input** and **output**

```
shinyServer(function(input, output){
  # Your server code goes here
})
```

input - Access user inputs on the server side
Example: input$some_user_input

output - Assign output like charts, tables, calculated values, etc.
Example: output$my_chart

Shiny does its reactive magic by defining your data as a reactive expression

```
# define your data as a reactive expression
myData <- reactive({
  df %>%
    filter(some_var > input$some_threshold)
})

# your variable “myData” is now a reactive
# expression. Call your data by running the reactive
# function
myData()
```

Let’s build a Shiny app

- I downloaded data on Allegheny County car crashes in 2015

- I built a simple Shiny app to explore the crash data which we will recreate together now

Resources

Shiny app we are building
https://github.com/dhenderson/shinyheinz

RStudio Shiny tutorial
http://shiny.rstudio.com/tutorial/

RStudio Shiny reference
http://shiny.rstudio.com/reference/shiny/latest/

Me
david@fii.org @david_henderson