Personal Management Tool

Members: Cheng Wang Xu Zhang Yang He

TaskLogger

Goal:

- Manage the daily tasks and events
- Flexible
- Categorize the tasks easily
- Generate reports

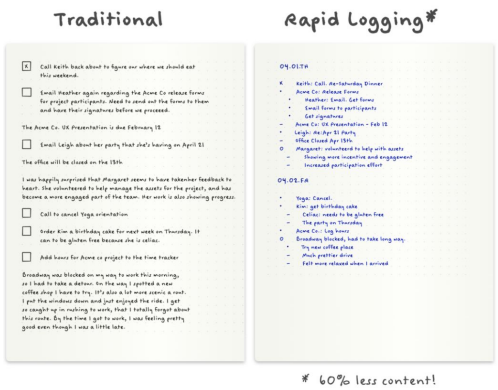
TaskLogger

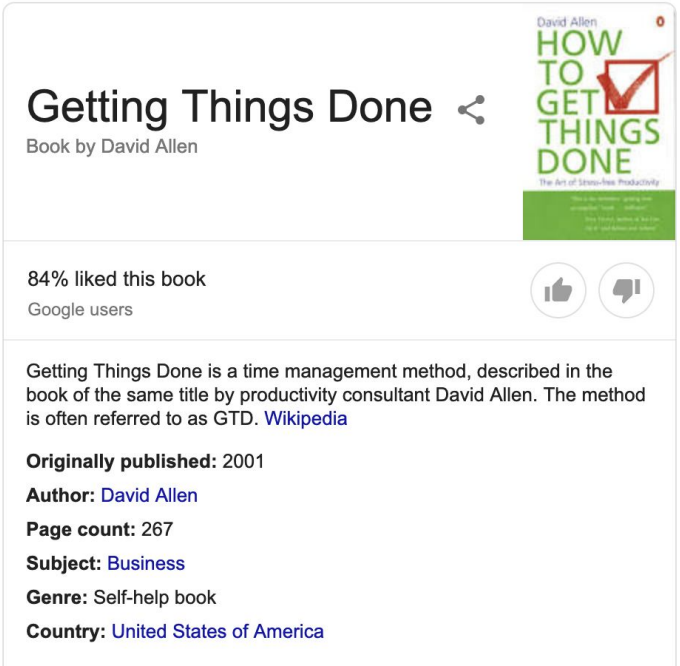
Some related Methodology

GTD:

"GTD is the work-life management system that has helped countless individuals and organizations bring order to chaos."

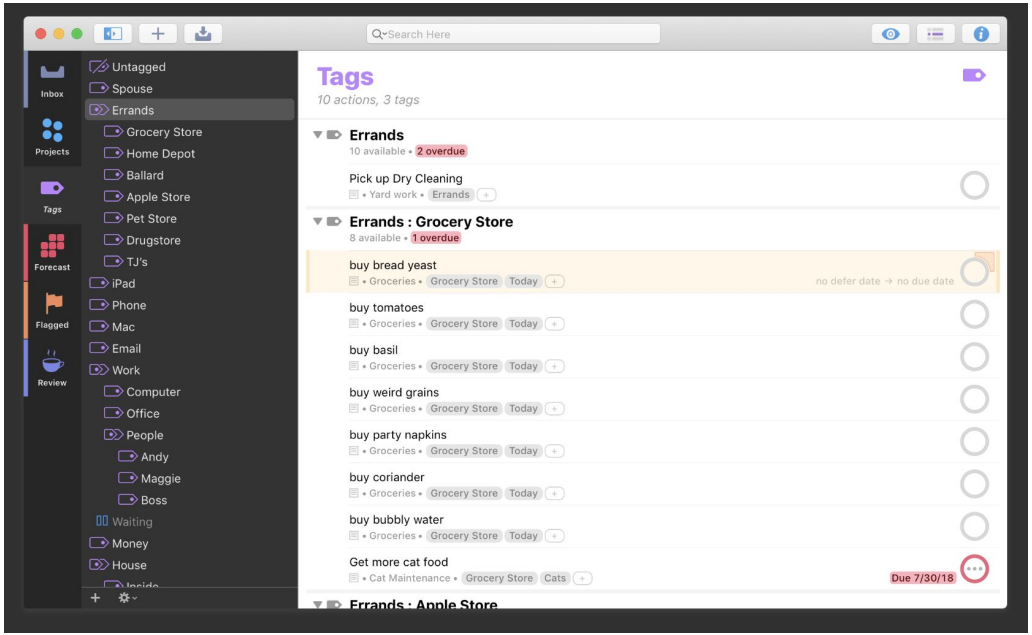
Bullet Journal



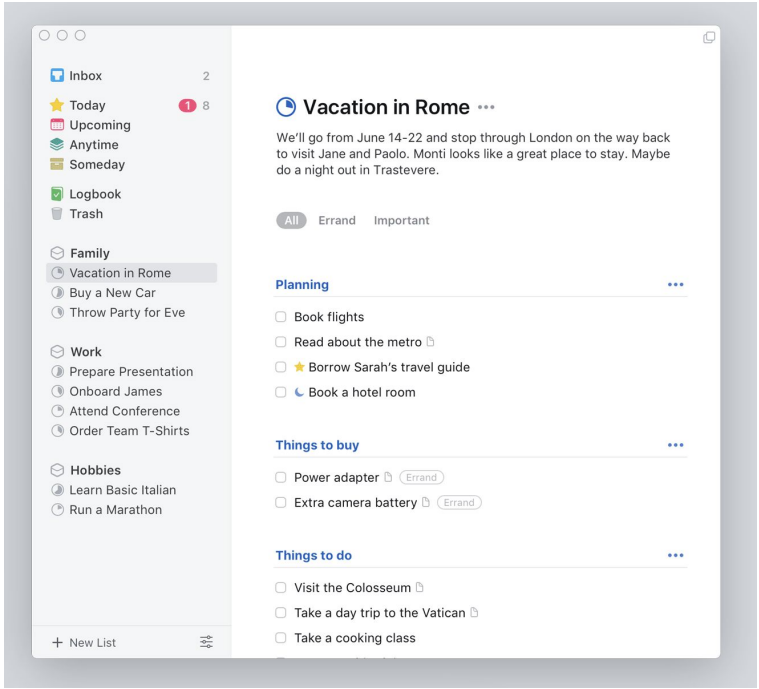


TaskLogger

Some softwares(tools) I’ve used



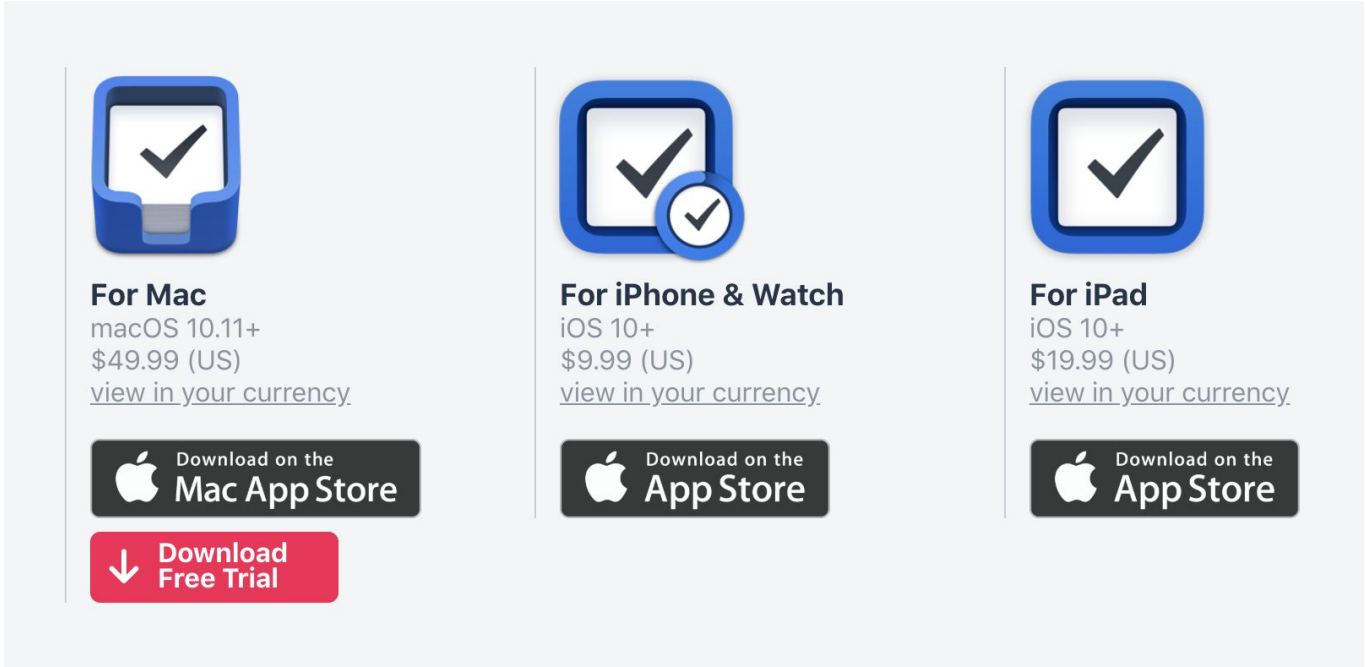
OmniFocus



Things 3

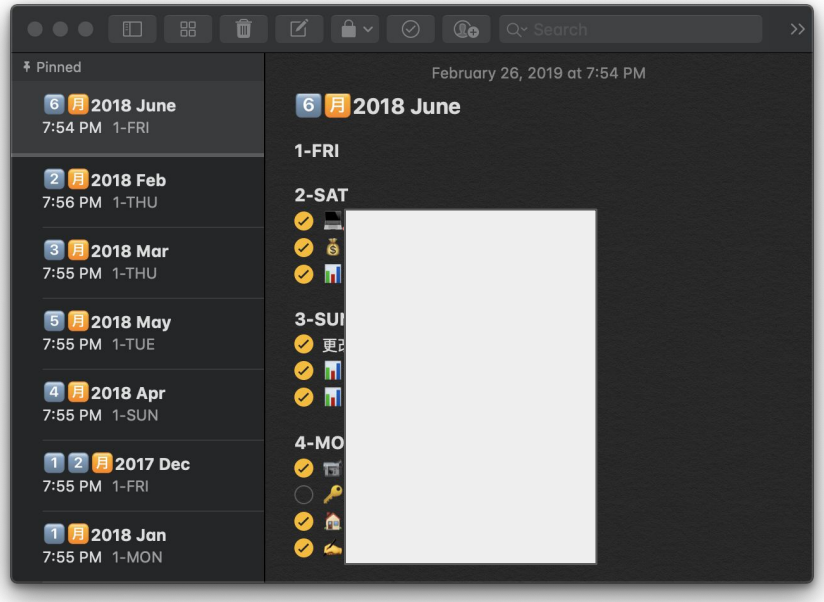
TaskLogger

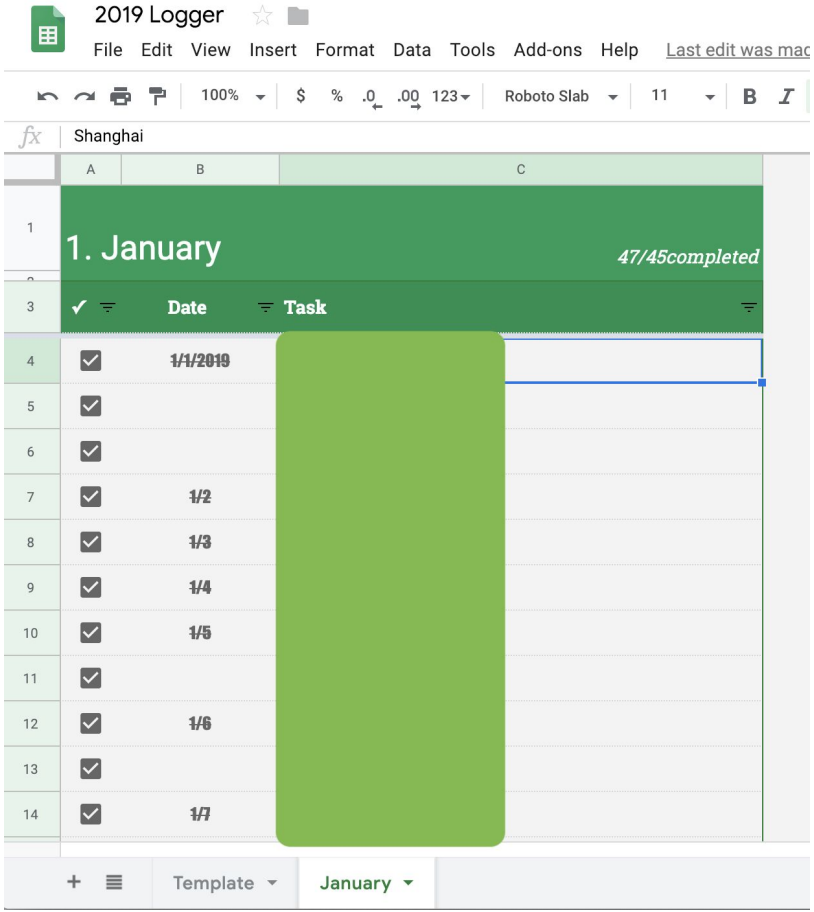
Some softwares(tools) I’ve used



TaskLogger

My Experiment





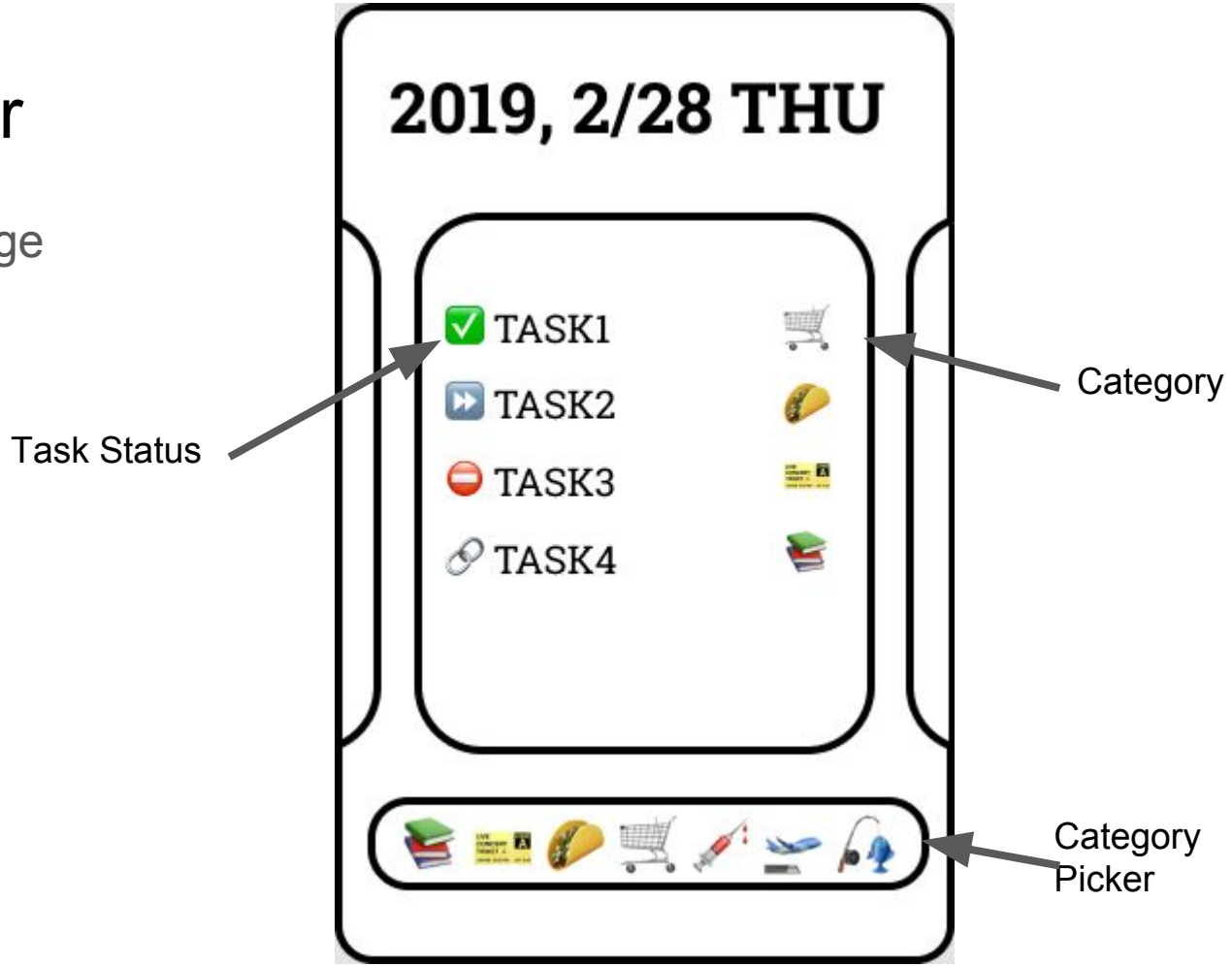
TaskLogger - The Website Structure

Routing

- Everyday page
- Report Page
- Export Page
- Setting Page
- About Page
- Product Page(sitting log)

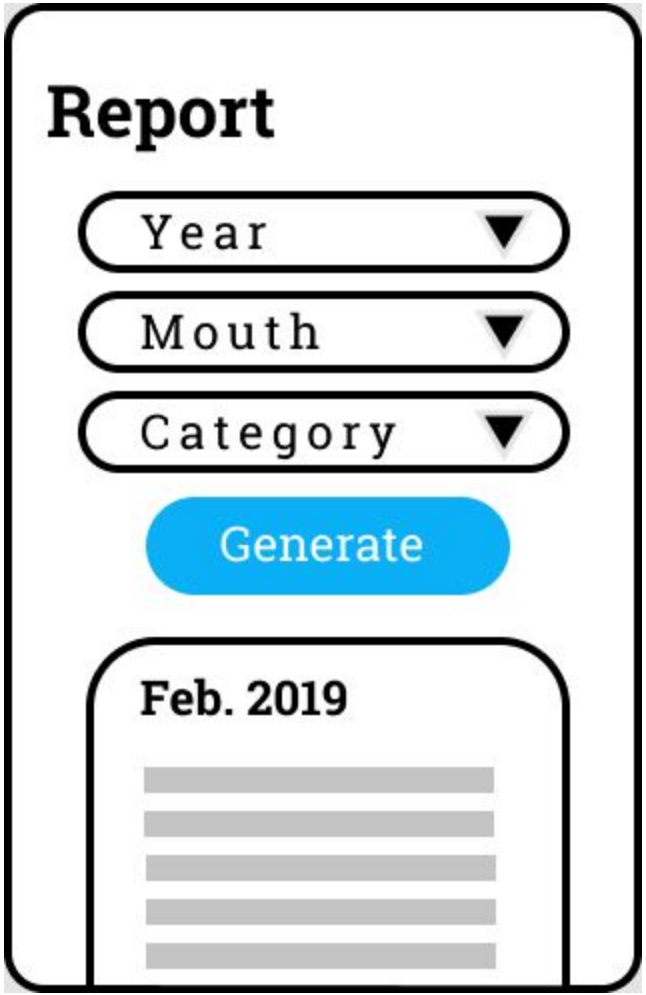
TaskLogger

Everyday page



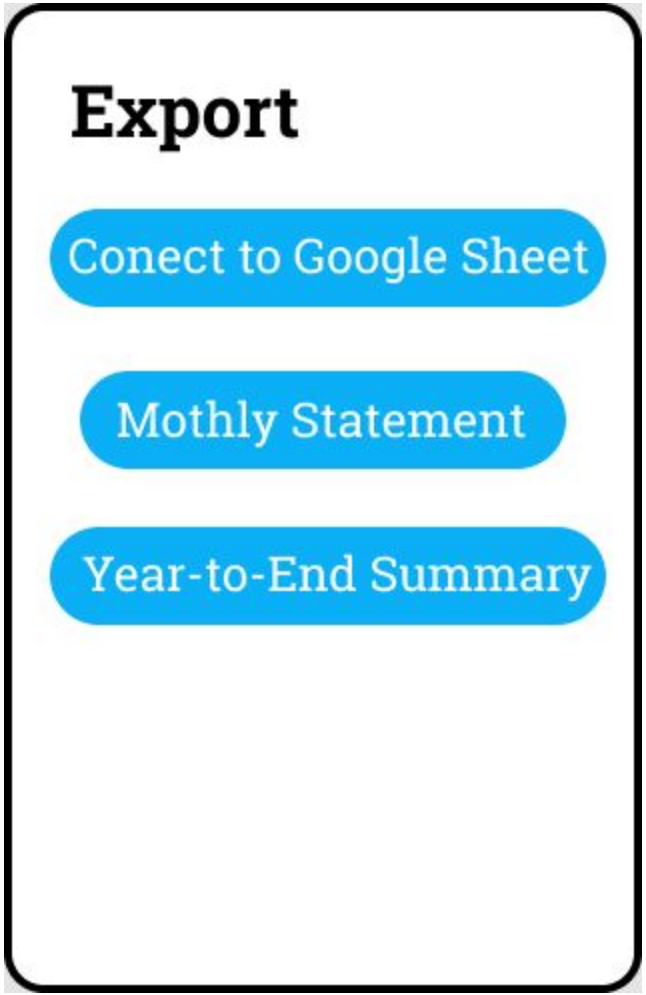
TaskLogger

Report page



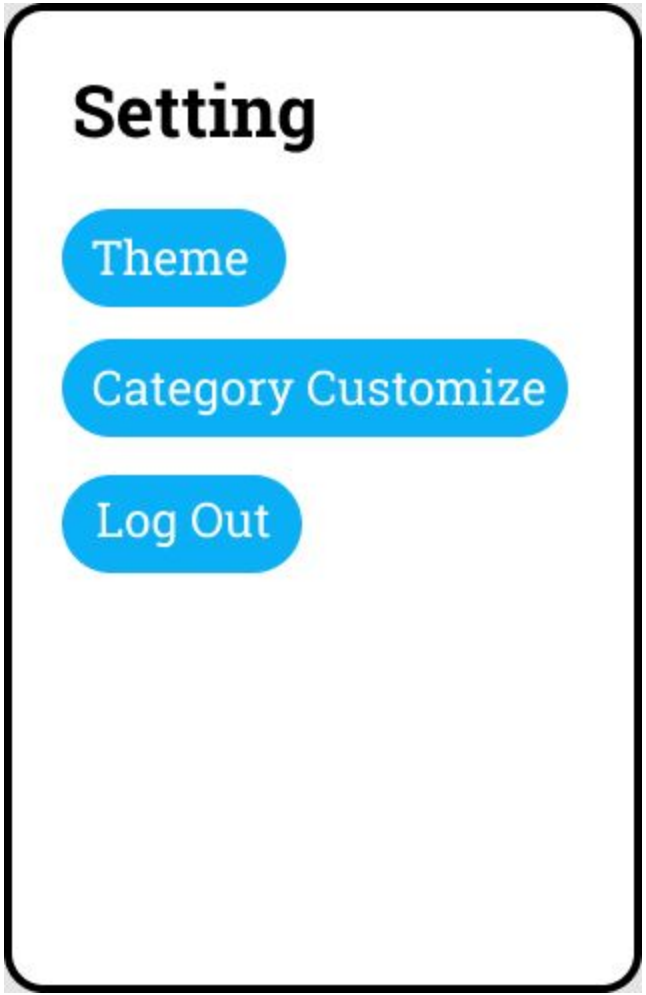
TaskLogger

Export page



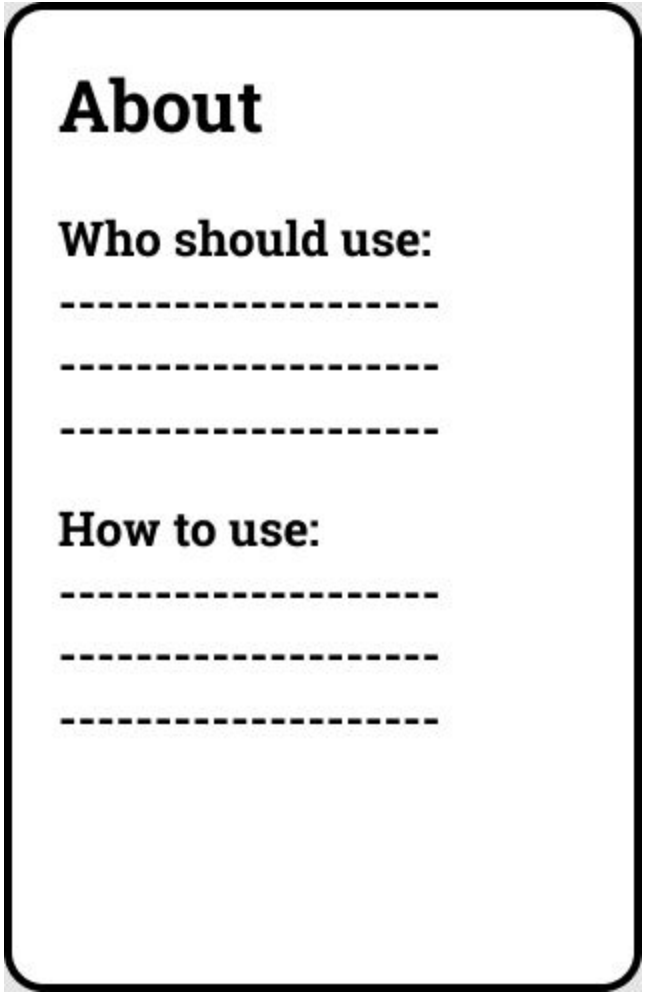
TaskLogger

Setting page



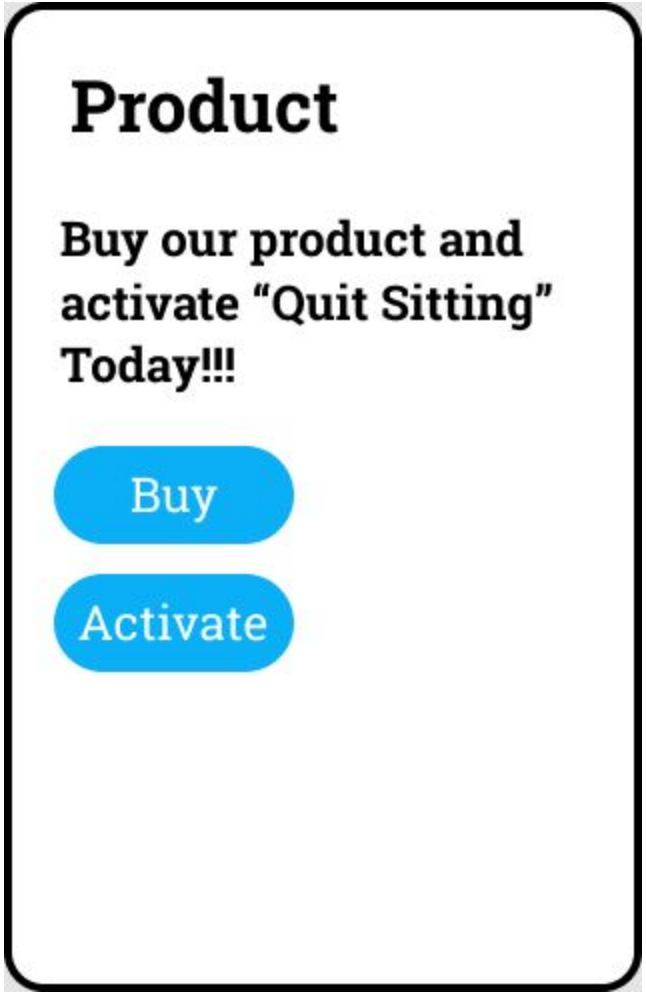
TaskLogger

About page



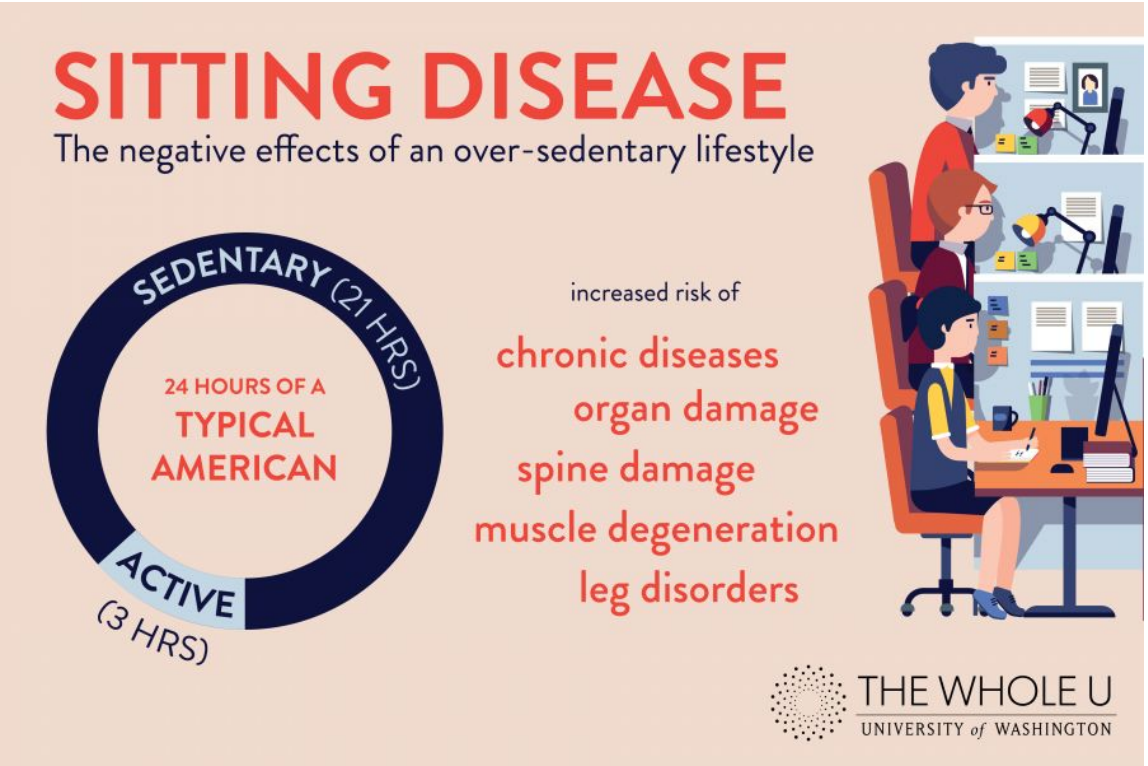
TaskLogger

Product page



Sitting logging webapp

Sitting is the new smoking!



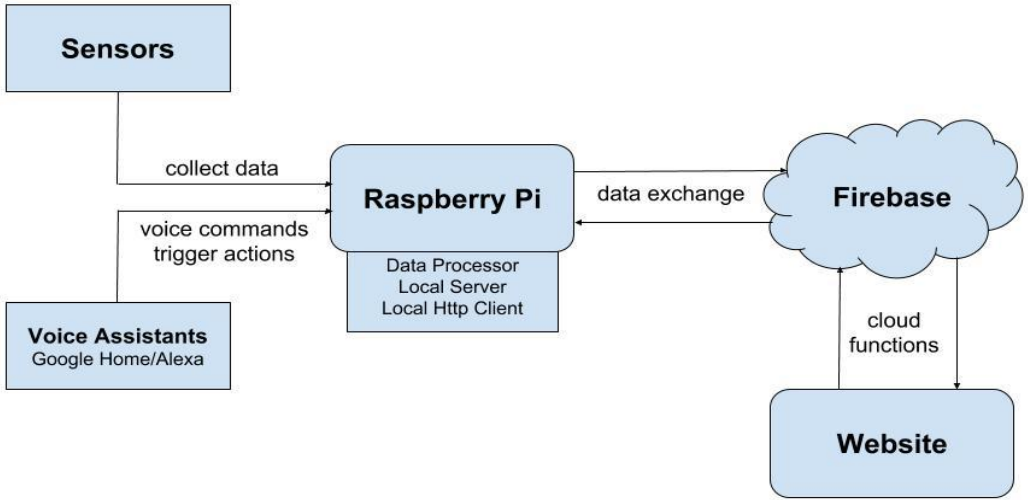
Project goals:

- Use **Raspberry Pi** to connect sensor to chair to track user sitting activity.
- Design a webapp to display the data, create notifications and generate logging.
- Add voice control using **Google Home** to enable interactions.

Uniqueness:

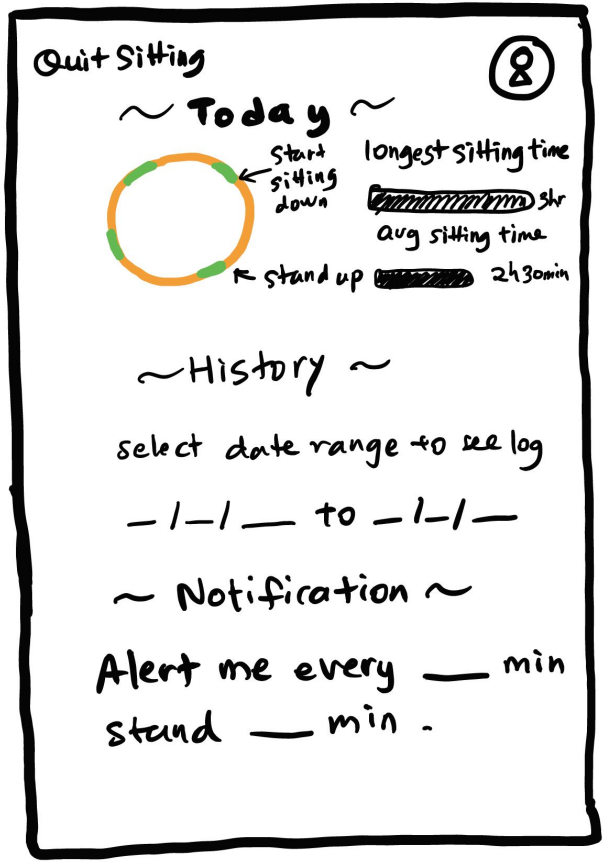
- Tracking automatically happens when user sit down
- Data privacy
- Cost-effective
- Voice-controlling is fun

Structure Diagram



Web design


Quit Sitting
HOME
USER
Why sitting is bad
login


Quit Sitting
~ Today ~
Start sitting down
longest sitting time
5hr
avg sitting time
stand up
2h30min
~History ~
select date range to see log
_/_/__ to _/_/__
~ Notification ~
Alert me every __ min
stand __ min.

Why combine our projects

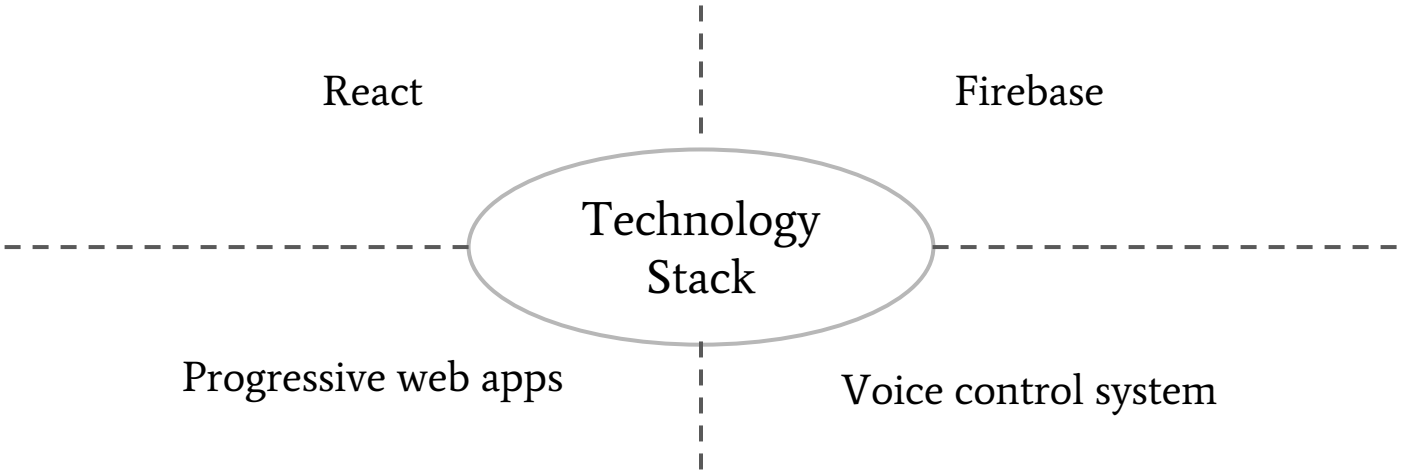
- Similar purpose

Task Logger	Sitting Logging
To provide user with a more efficient and convenient tool to manage daily tasks.	To develop a program to track users' sitting activity and visualize sitting data for users.

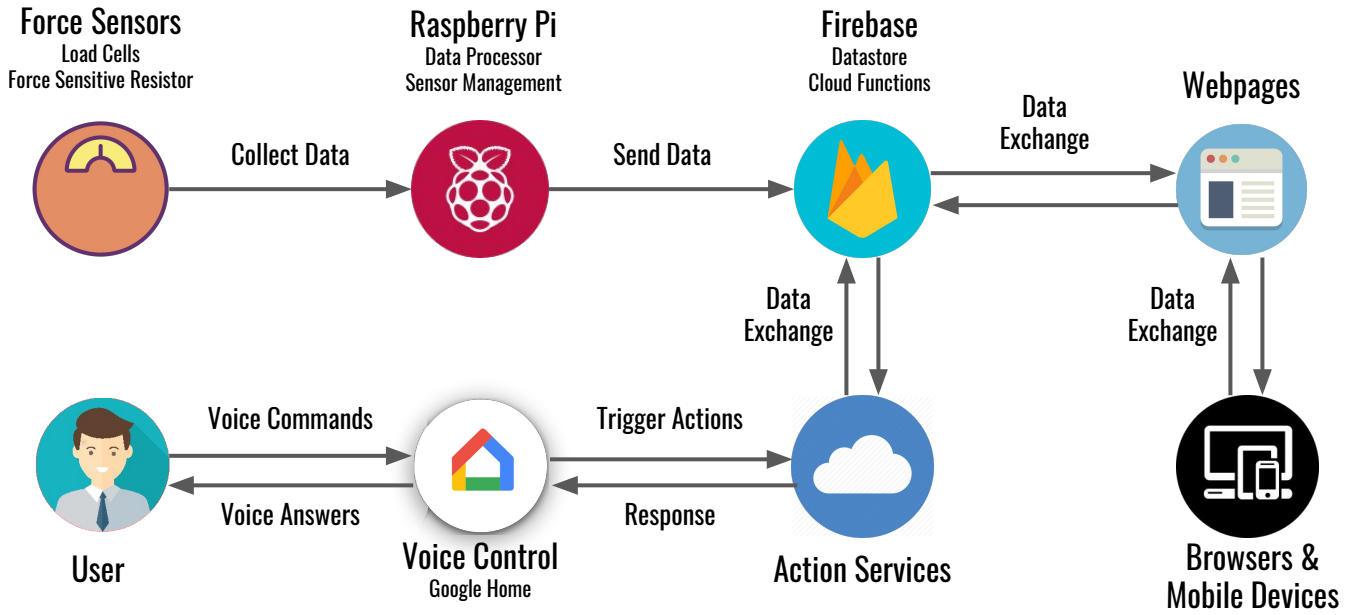
To build a tool to help users manage their personal data

Why combine our projects

- Similar tech stack



Diagram


Force Sensors
Load Cells
Force Sensitive Resistor
Collect Data
Raspberry Pi
Data Processor
Sensor Management
Send Data
Firebase
Datastore
Cloud Functions
Data
Exchange
Webpages
Data
Exchange
Data
Exchange
Voice Commands
Voice Answers
User
Voice Control
Google Home
Trigger Actions
Response
Action Services
Browsers &
Mobile Devices

Technologies

Technology Overview

- Raspberry Pi and Sensors
- Voice Control System
- React
- Progressive Web App (PWA)
- Firebase

Intro to
Raspberry Pi and Sensors

Hey, Raspberry Pi

- The Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote teaching of basic computer science in schools and in developing countries.
- You can use it to learn coding and to build electronic projects, and for many of the things that your desktop PC does, like spreadsheets, word processing, browsing the internet, and playing games.

Raspberry is a reference to a fruit naming tradition in the old days of microcomputers.

Pi is because originally we were going to produce a computer that could only really run Python. So the Pi in there is for Python.

What is the story behind the name "Raspberry Pi"? - Quora

Using Raspberry Pi - Python Requests

- Requests is an elegant and simple HTTP library for Python, built for human beings.
- It is based on http, urllibs and some other Python’s standard libraries but provides simpler methods to construct requests and process response messages.

Using Raspberry Pi - Python Requests

Request

- requests.get(), requests.post(), …

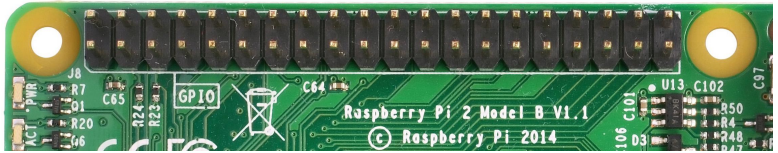
Response

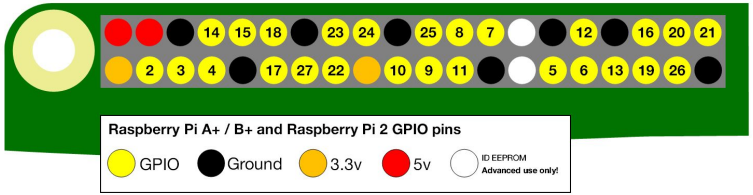
- By property: url, headers, text, raw, …
- By methods: json(), ...

```
pi@raspberrypi:~ $ python
Python 2.7.13 (default, Sep 26 2018, 18:42:22)
[GCC 6.3.0 20170516] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import requests
>>> resp=requests.post("http://httpbin.org/post", data={'key':'test1'})
>>> resp.url
u'http://httpbin.org/post'
>>> resp.headers
{'Content-Length': '264', 'Content-Encoding': 'gzip', 'Server': 'nginx',
'Connection': 'keep-alive', 'Access-Control-Allow-Credentials': 'true', '
Date': 'Tue, 05 Mar 2019 13:40:31 GMT', 'Access-Control-Allow-Origin': '*
', 'Content-Type': 'application/json'}
>>> resp.text
u'{\n  "args": {}, \n  "data": "", \n  "files": {}, \n  "form": {\n    "k
ey": "test1"\n  }, \n  "headers": {\n    "Accept": "*/*", \n    "Accept-E
ncoding": "gzip, deflate", \n    "Content-Length": "9", \n    "Content-Ty
pe": "application/x-www-form-urlencoded", \n    "Host": "httpbin.org", \n
    "User-Agent": "python-requests/2.12.4"\n  }, \n  "json": null, \n  "o
rigin": "72.230.83.162, 72.230.83.162", \n  "url": "https://httpbin.org/p
ost"\n}\n'
>>> resp.json()
{u'files': {}, u'origin': u'72.230.83.162, 72.230.83.162', u'form': {u'ke
y': u'test1'}, u'url': u'https://httpbin.org/post', u'args': {}, u'header
s': {u'Content-Length': u'9', u'Accept-Encoding': u'gzip, deflate', u'Acc
ept': u'*/*', u'User-Agent': u'python-requests/2.12.4', u'Host': u'httpbi
n.org', u'Content-Type': u'application/x-www-form-urlencoded'}, u'json':
None, u'data': u''}
```

Using Raspberry Pi - GPIO

- General Purpose Input/Output on the RPi
- Two 5V pins and two 3V3 pins are present on the board, as well as a number of ground pins (0V), which are unconfigurable. The remaining pins are all general purpose 3V3 pins, meaning outputs are set to 3V3 and inputs are 3V3-tolerant.
- Inputs and Outputs: high (3V3) or low (0V)
- Protocols: PWM, SPI, I2C, and Serial





Using Raspberry Pi - GPIO

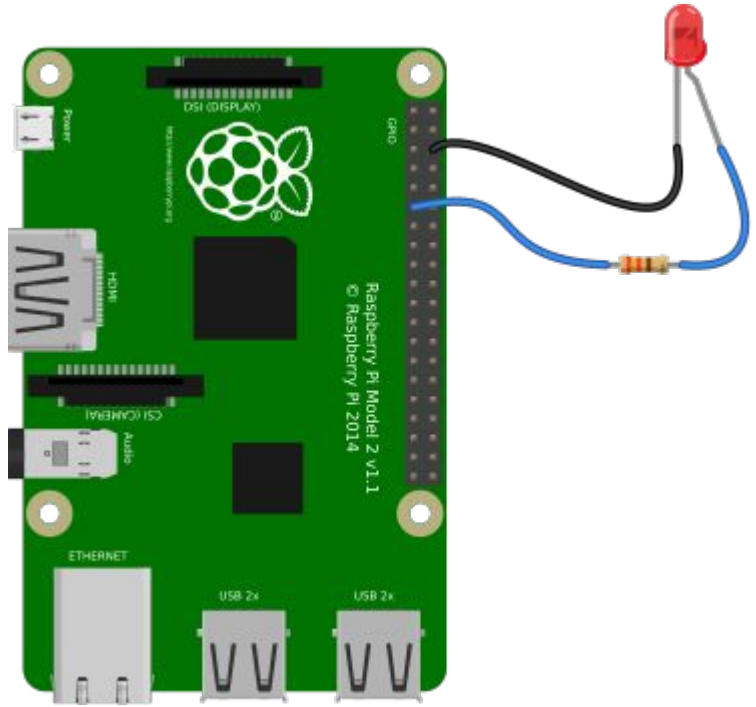
- GPIO library: RPi.GPIO
- Code example:

```
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

GPIO.setup(17, GPIO.OUT)

GPIO.output(17, GPIO.HIGH)
```



Source: https://gpiozero.readthedocs.io/en/stable/recipes.html

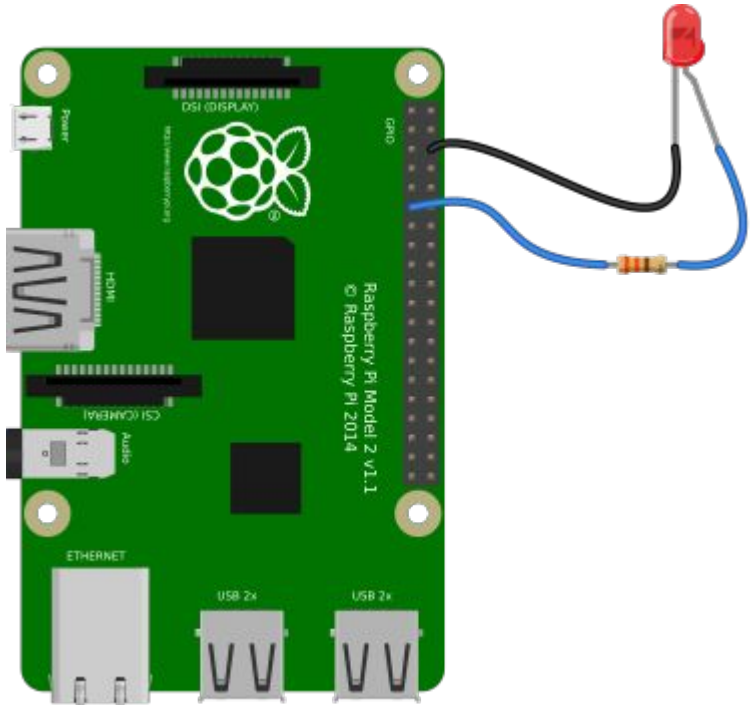
Using Raspberry Pi - GPIO

- GPIO library: gpiozero
- A layer on top of RPi.GPIO
- Code example:

```
from gpiozero import LED
from time import sleep

led = LED(17)

while True:
    led.on()
    sleep(1)
    led.off()
    sleep(1)
```



Source: https://gpiozero.readthedocs.io/en/stable/recipes.html

Sensors

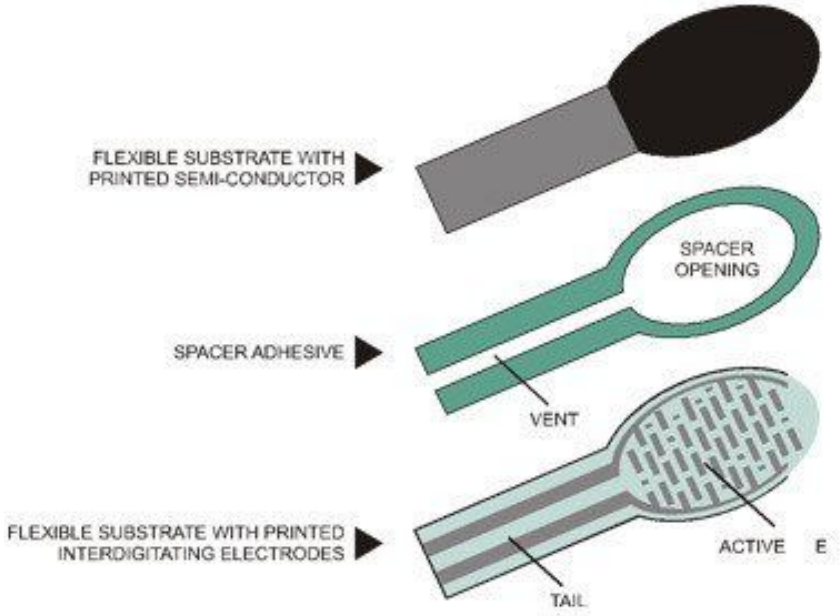
- Force sensitive resistor

  Used to detect physical pressure, squeezing, and weight

- Half-bridge strain gauge load cell

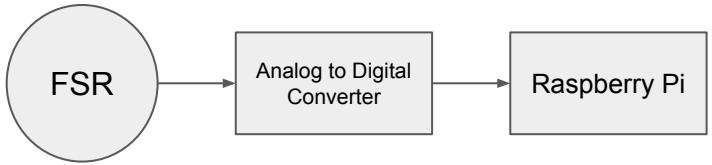
  Widely used in bathroom scales

Sensors - Force sensitive resistors

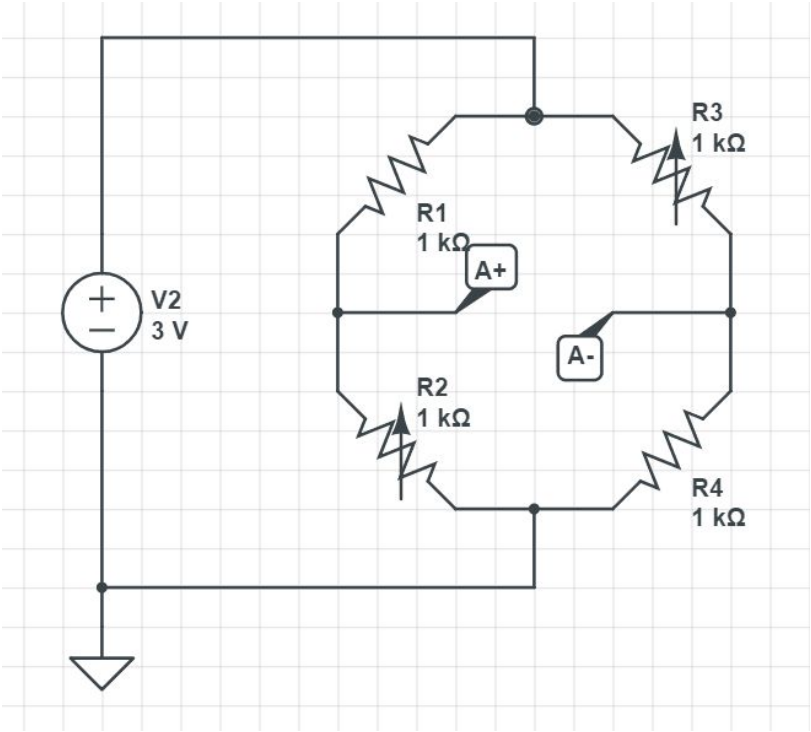


Source: https://learn.adafruit.com/force-sensitive-resistor-fsr/overview

- FSRs are basically a resistor that changes its resistive value (in ohms Ω) depending on how much it is pressed.
- The FSR is made of 2 layers separated by a spacer. The more one presses, the more of those Active Element dots touch the semiconductor and that makes the resistance go down.



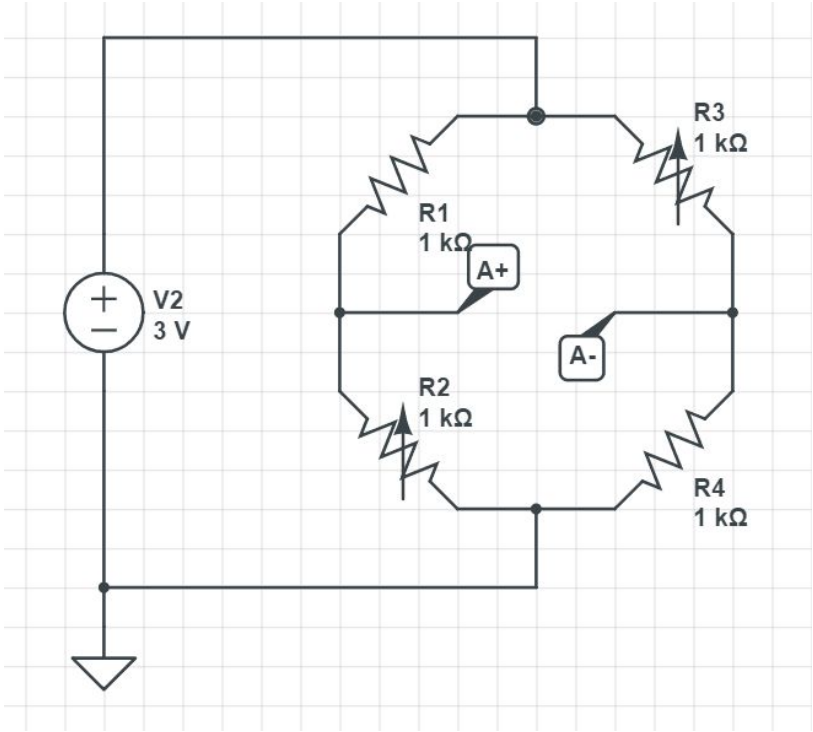
Sensors - Half-bridge strain gauge load cell



- Strain Gauge

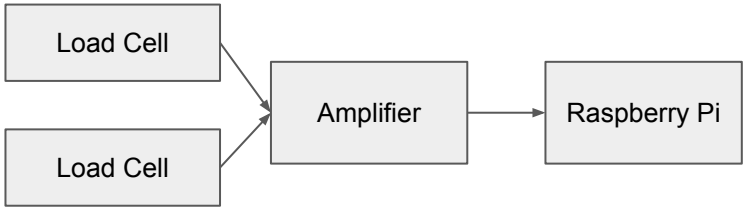
  When an electrical conductor is stretched within the limits of its elasticity such that it does not break or permanently deform, it will become narrower and longer, which increases its electrical resistance end-to-end.

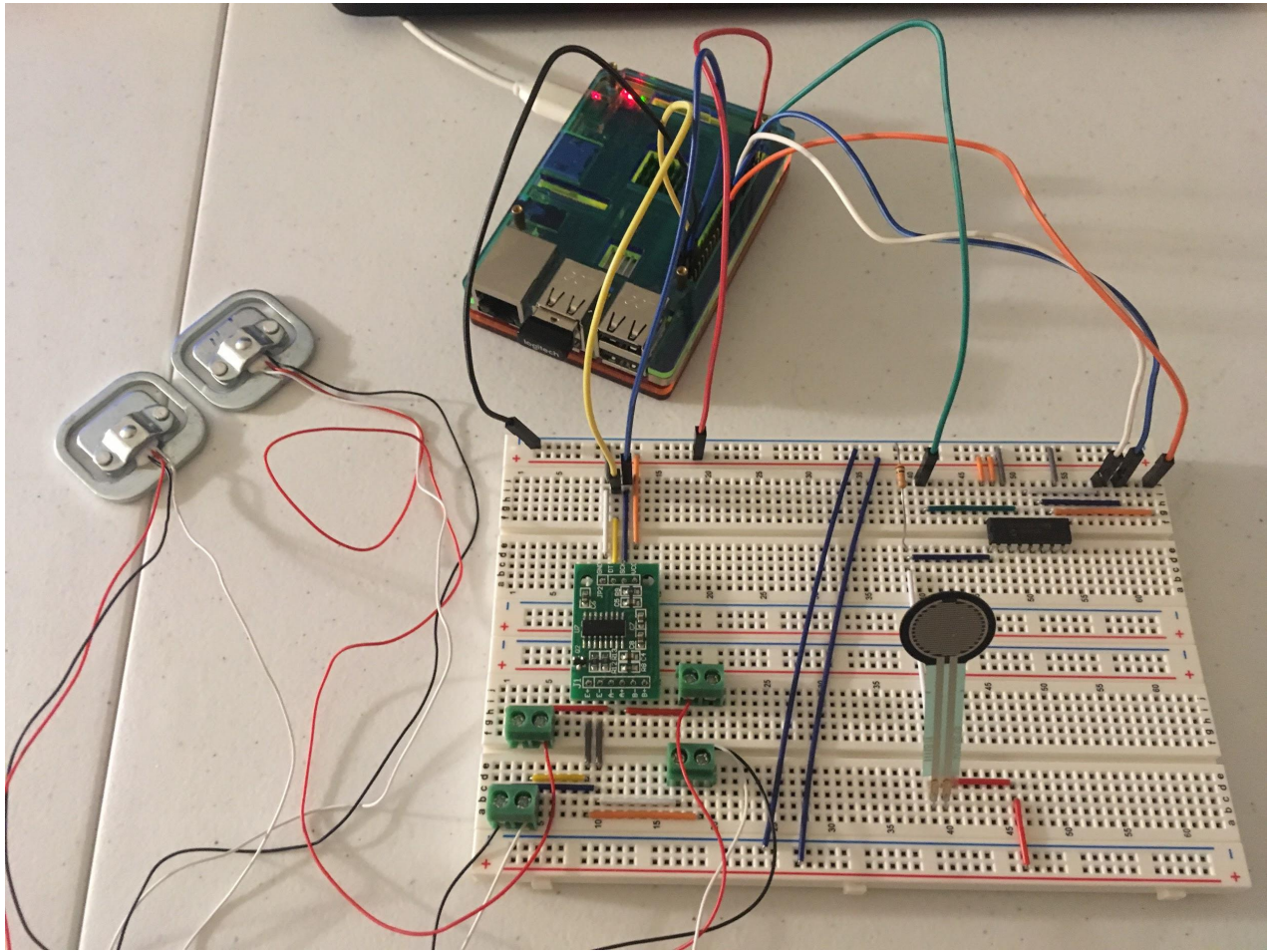
Sensors - Half-bridge strain gauge load cell



- Wheatstone bridge

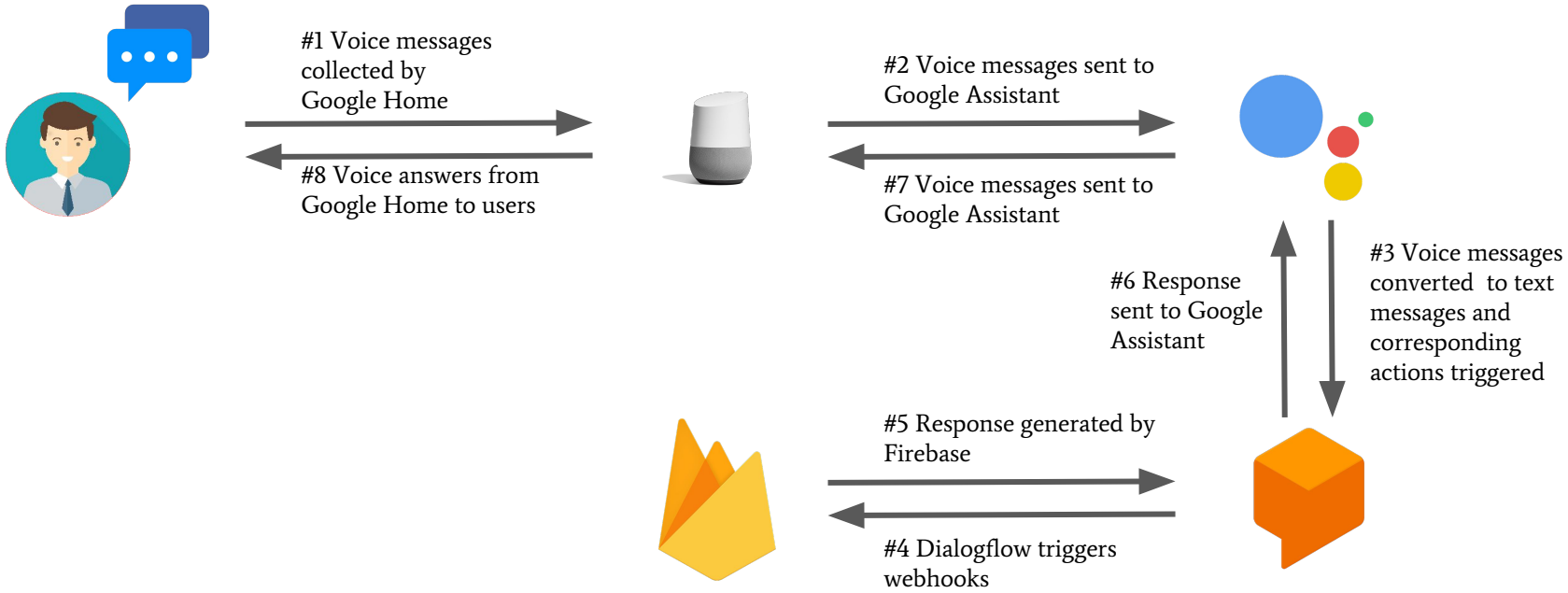
  A Wheatstone bridge is an electrical circuit used to measure an unknown electrical resistance by balancing two legs of a bridge circuit, one leg of which includes the unknown component.





Intro to
Voice Control System

Diagram


#1 Voice messages collected by Google Home
#8 Voice answers from Google Home to users
#2 Voice messages sent to Google Assistant
#7 Voice messages sent to Google Assistant
#3 Voice messages converted to text messages and corresponding actions triggered
#6 Response sent to Google Assistant
#5 Response generated by Firebase
#4 Dialogflow triggers webhooks

Hey, Google - Google Home

Smart speaker

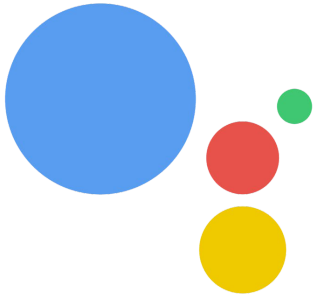
A type of wireless speaker and voice command device with an integrated virtual assistant that offers interactive actions and hands-free activation with the help of one "hot word" (or several "hot words").

Google Home

A smart speaker that enables users to speak voice commands to interact with services through Google's personal assistant software called Google Assistant.

Google Assistant

An artificial intelligence-powered virtual assistant developed by Google that is primarily available on mobile and smart home devices.



hey google

Hi, what can I do for you?

i'm a robot

01001000 01101001. Just joking. That means "Hi" in binary code 😎

Actions and Dialogflow

Actions on Google

A platform for developers to extend the Google Assistant by implementing customized actions.

Dialogflow

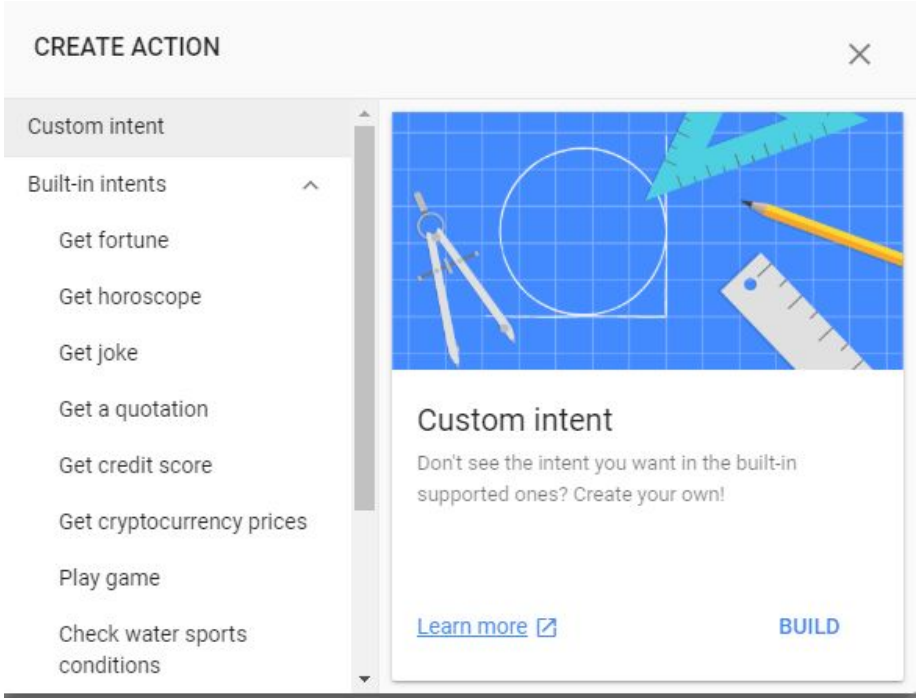
Give users new ways to interact with your product by building engaging voice and text-based conversational interfaces, such as voice apps and chatbots, powered by AI.

Actions and Dialogflow

Actions

An interaction you build for the Assistant that supports a specific intent and has a corresponding fulfillment that processes the intent.

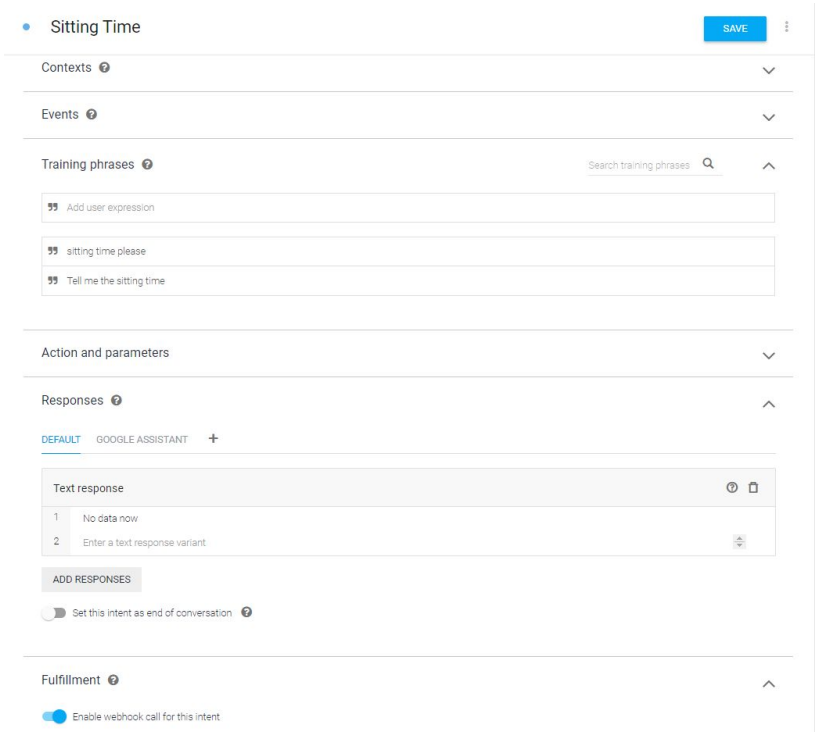
- Smart home Actions
- Conversational Actions



Actions and Dialogflow

Intent

A goal or task that users want to do, such as ordering coffee or finding a piece of music. In Actions on Google, this is represented as a unique identifier and the corresponding user utterances that can trigger the intent.

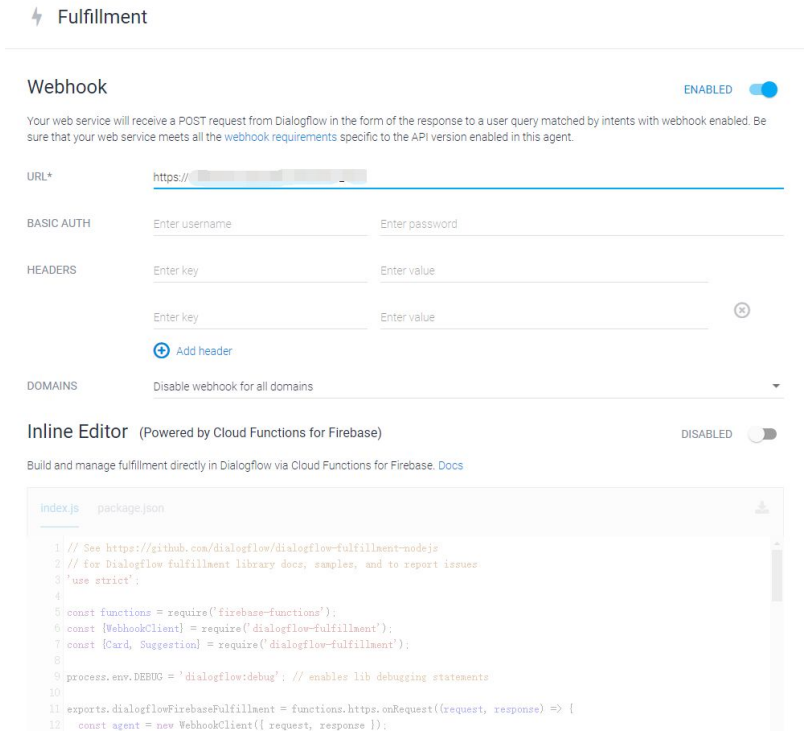


Actions and Dialogflow

Fulfillment

A service, app, feed, conversation, or other logic that handles an intent and carries out the corresponding Action.

- By the inline editor
- By webhooks

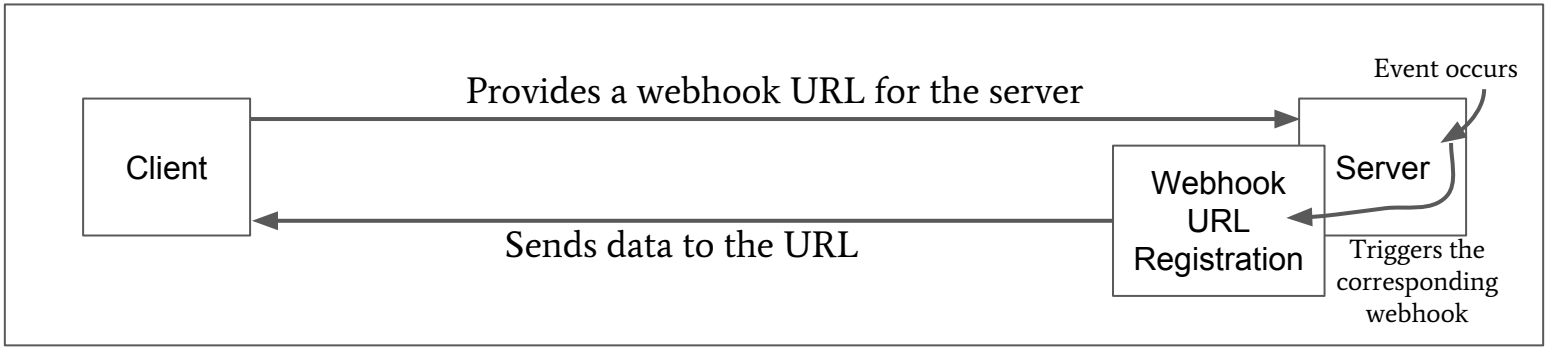


Webhook

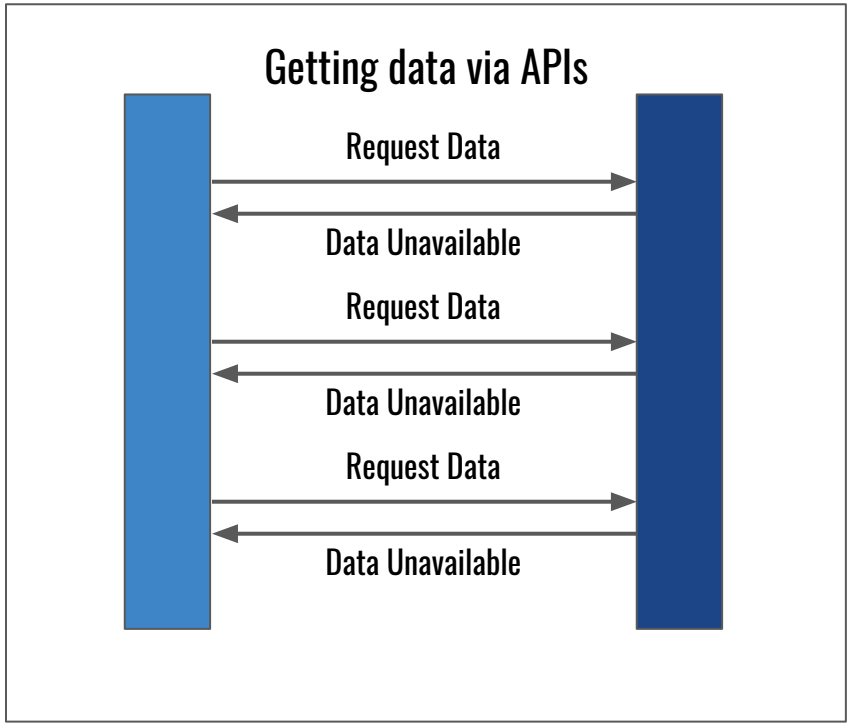
- An HTTP callback

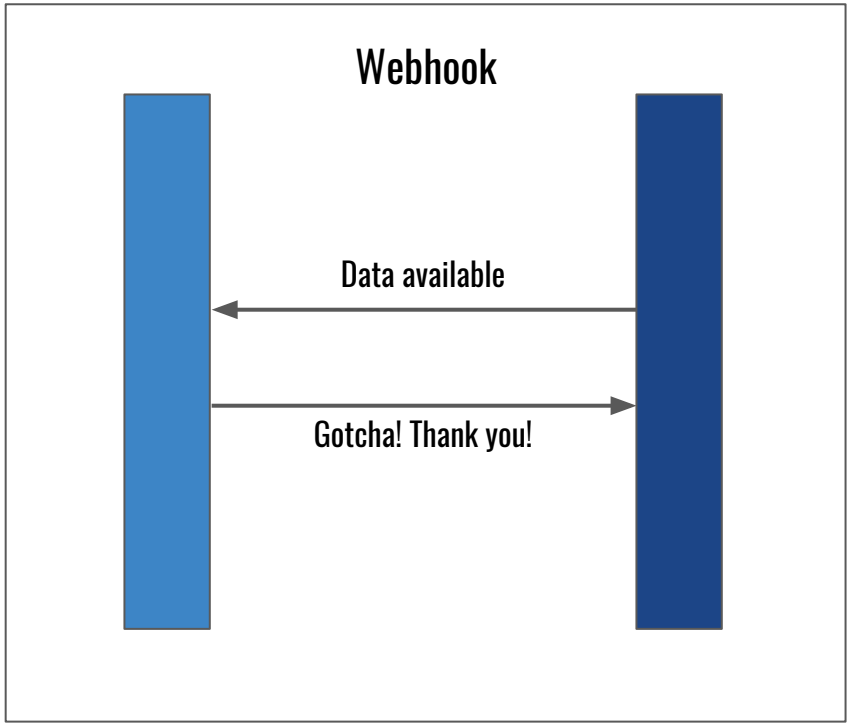
  It is a simple notification sent via HTTP POST when certain events happen.

- Reverse API
- Examples: Github webhooks, Slack webhooks, ...



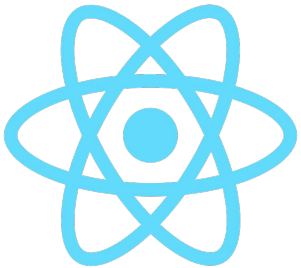
Webhooks vs API





Intro to React.js

What is React.js?



- "A Javascript library for creating user interfaces"
- "The 'V' in 'MVC'"
- "A library, not a framework"

What is React.js?

Declarative

- you can build Web interfaces without even touching the DOM directly
- you can have an event system without having to interact with the actual DOM Events.
- In contrast with JQuery

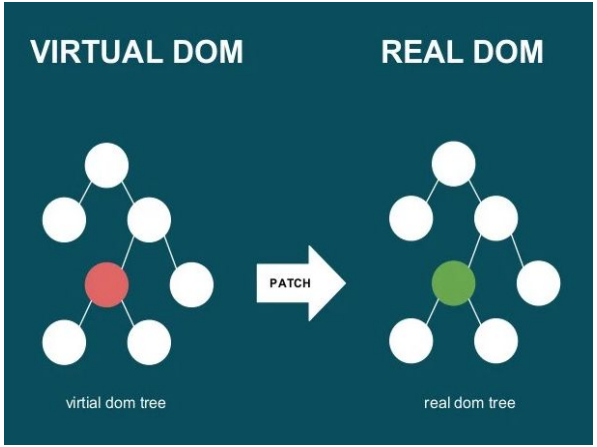
Component-based

- A method for breaking down larger UI interfaces into independent, self-sustaining micro-systems.
- Built off of the concept of Ajax requests.
- Reusability, single-responsibility

What is React.js?

- **Virtual DOM**

  A virtual DOM object is a representation of a DOM object, like a lightweight copy.



Source: https://www.vishwainfoways.com/blog/difference-between-dom-and-virtual-dom/

- **JSX, aka JavaScript eXtension**

```
const element = <h1>Hello, world!</h1>;
```

Source: https://twitter.com/iamdevloper/status/598435575662813184

React Ecosystem

- **Common Libraries:**

1. Routing: **React-Router**
2. AJAX requests: **Axios**, SuperAgent
3. State Management: **Redux**
4. Native apps: **React-Native**
5. Set up: **create-react-app**, Next.js, Gatsby.js

- **Build Tools:**

1. **Babel**: transforms JavaScript ES* (i.e., JS 2016, 2017) to ES5
2. **Webpack**: module loader and bundler

React Component

React component

Button was clicked:
4 times
Click Me

```
class Counter extends React.Component {
    state = {counter : 0}

    onClick = () => {
        this.setState({counter : this.state.counter + 1});
    }

    render() {
        const {counter} = this.state;

        return (
            <div>
                Button was clicked:
                <div>{counter} times</div>

                <button onClick={this.onClick}>Click Me</button>
            </div>
        );
    }
}
render(<Counter />, mountNode);
```

React component

- Functional Component

```
function FunctionalComponentSyntax(props) {
    return (
        <div>
            Hello World!
        </div>
    );
}
```

- Class Component

```
class Hello extends Component{
    render(){
        return <div>Hello {this.props.name}</div>
    }
}
```

Props vs. state

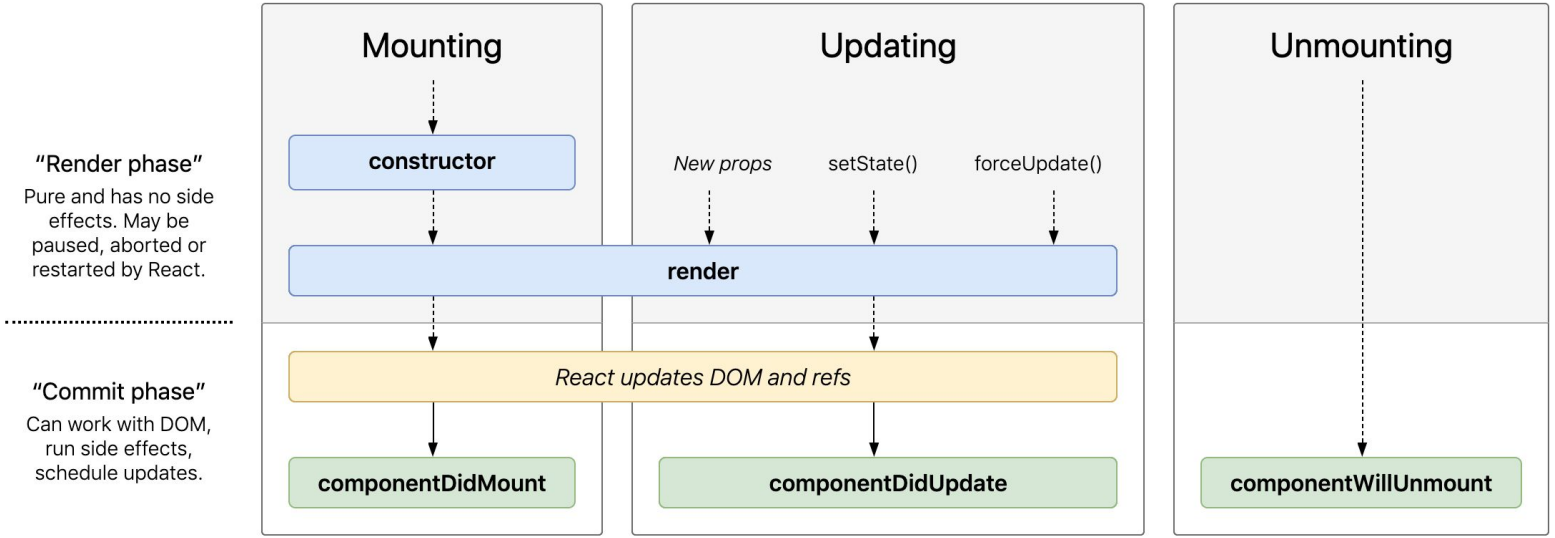
- State:

1. Local data of the component, no access outside the component.
2. Mutable
3. Use `SetState()` when changing the states.

- Props:

1. Receive data from parent component
2. Can have default values defined in the component
3. Should not change during the component’s life cycle
4. Can specify types of props using propType.

Life cycle of a component

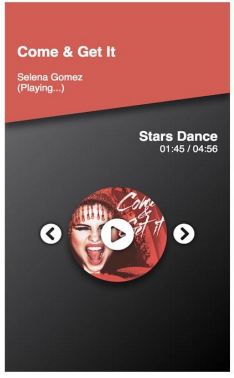


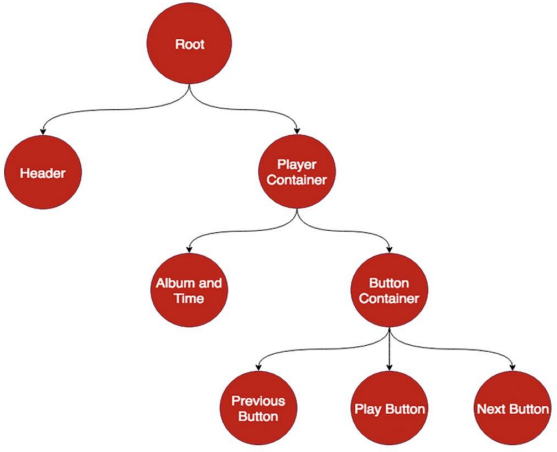
Source: http://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/

Context api: Why?

- Pass down data through the component tree without having to pass props down manually at every level.
- Like “global object”
- Examples of use case:

1. Themes
2. Multilingual application
3. Authentication





Source: https://blog.bitsrc.io/why-you-should-consider-the-new-context-api-in-react-a-deep-dive-d588b66c57b5

Context api: How?

```
const { Provider, Consumer } = React.createContext()
```

```
const App = () => {
  return (
    <userContext.Provider value={user}>
      <div>
        <Header>
          <Navbar />
        </Header>
        <Body>
          <SideBar>
            <Profile />
          </SideBar>
        </Body>
      </div>
    </userContext.Provider>
  );
};
```

```
const Profile = () => {
  return (
    <userContextConsumer>
      {value => (
        <div>
          <ProfileImage path={value.image_url} />
          <ProfileName name={value.name} />
        </div>
      )}
    </userContextConsumer>
  );
};
```

Resources for React

- Official Documentation:

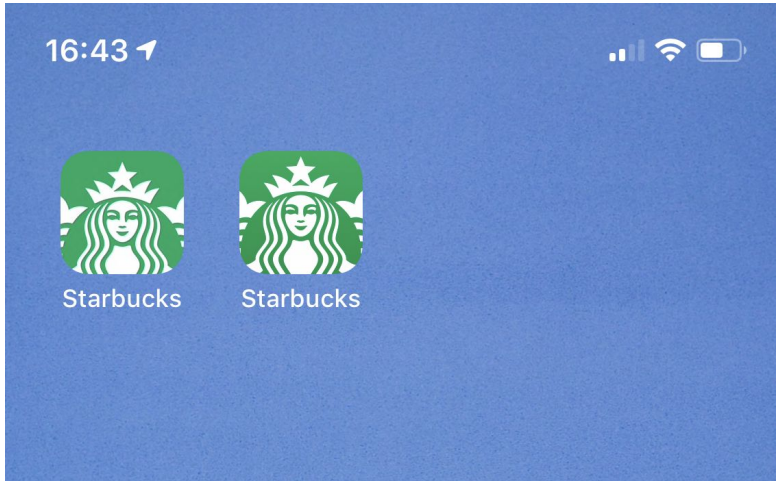
1. React.js
2. Create-react-app:

- Additional resources:

1. Curated React, Redux, ES6 links glossary
2. react enlightenment
3. 30 days of react

Progressive Web App

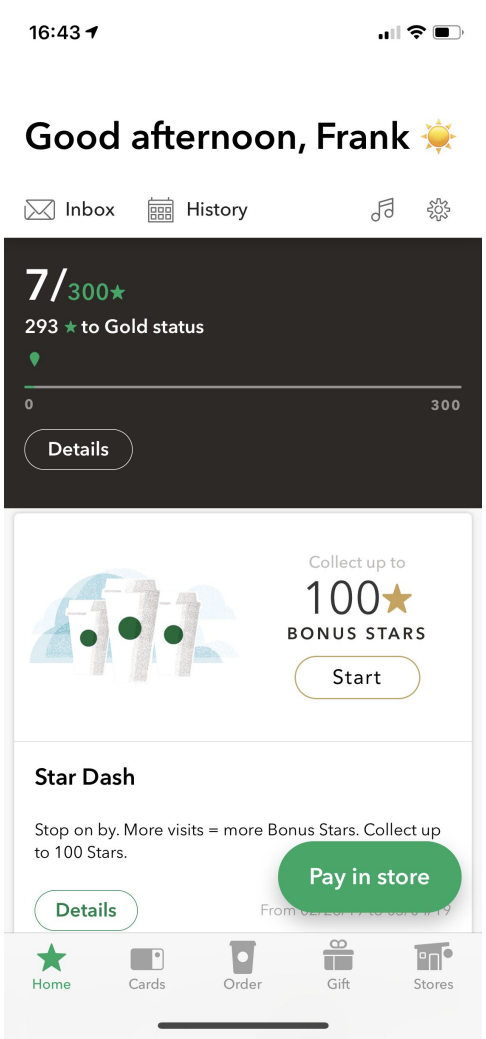
Example: Starbucks

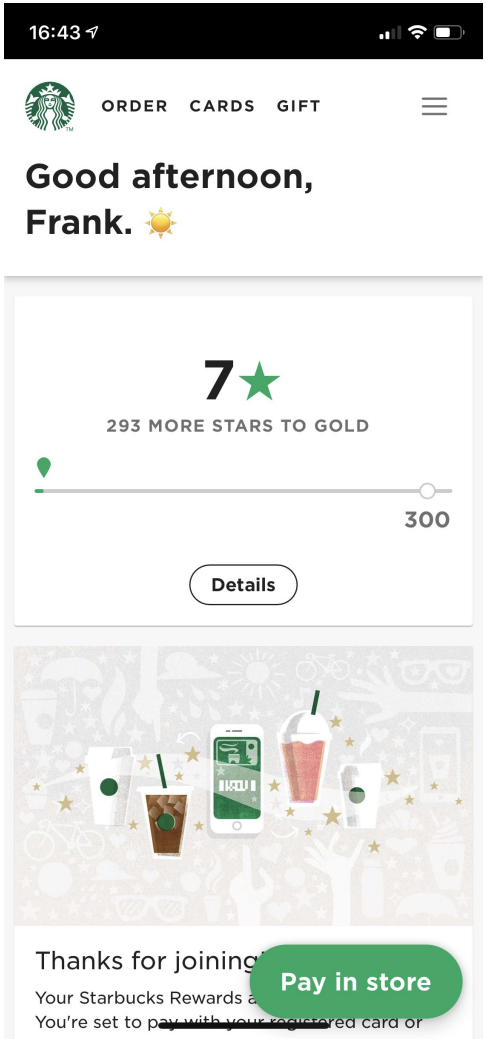


PWA

Example:

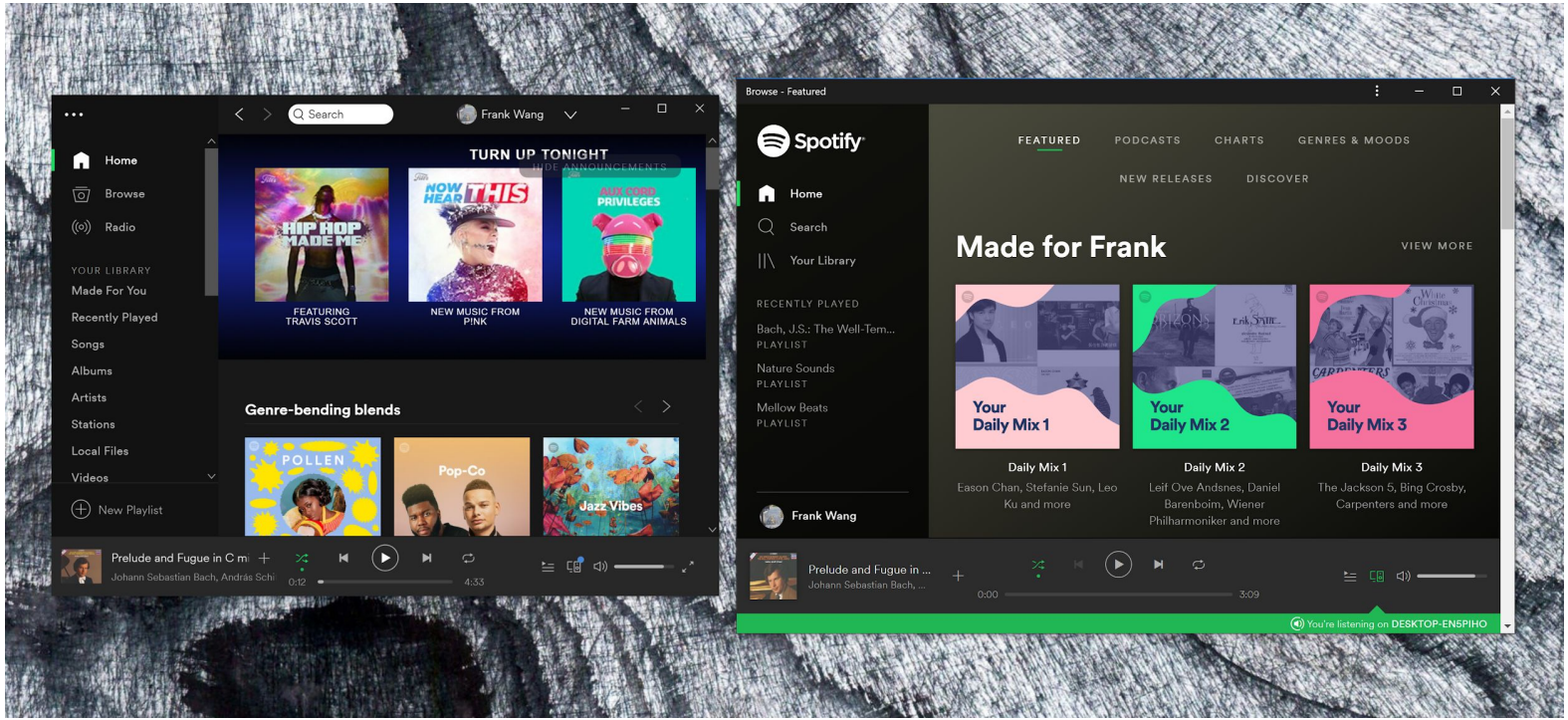
Starbucks





Progressive Web App

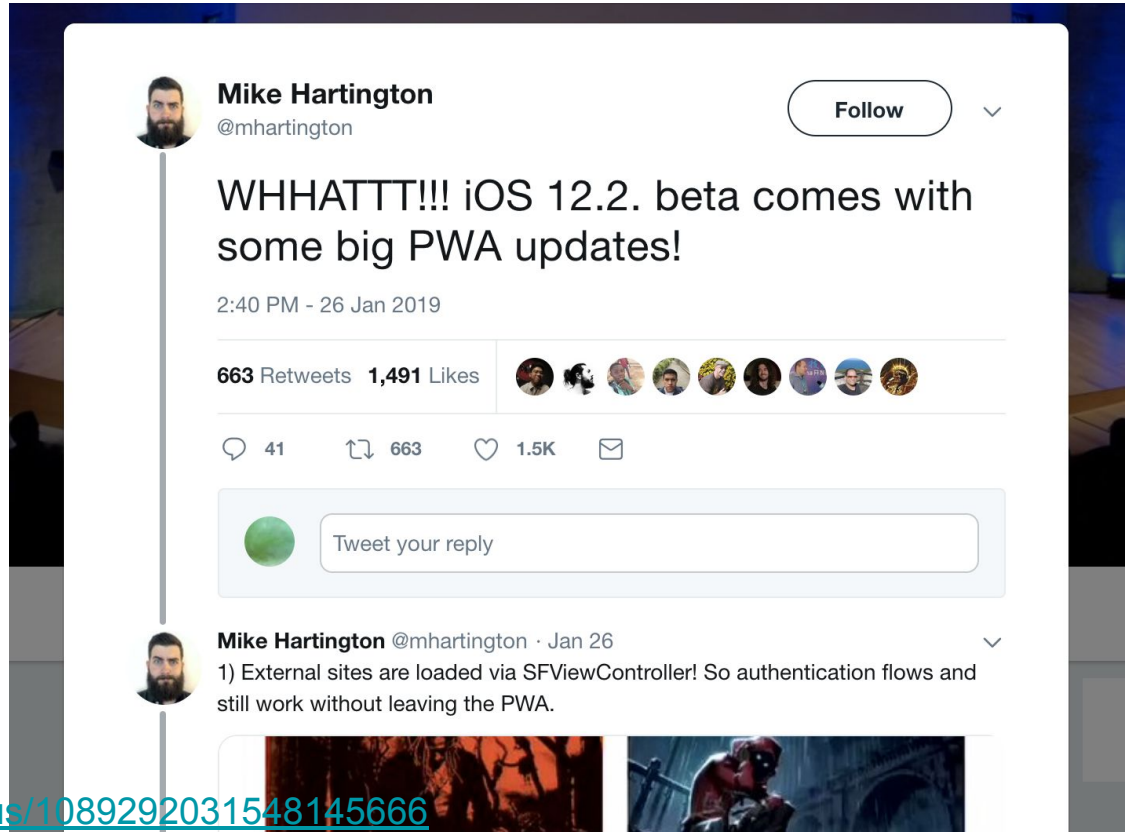
Example: Spotify on Windows



Progressive Web App

News on main OS:

iOS

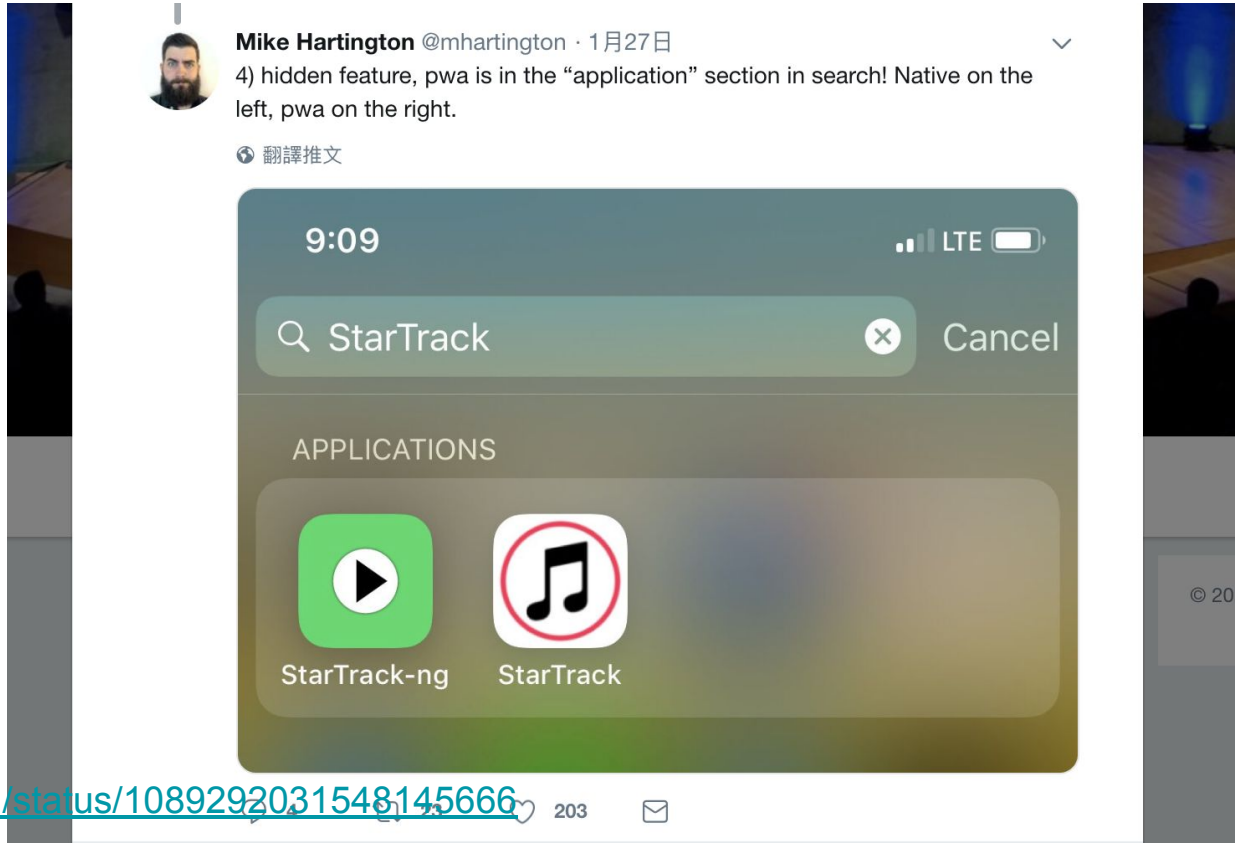


https://twitter.com/mhartington/status/1089292031548145666

Progressive Web App

News on main OS:

iOS



https://twitter.com/mhartington/status/1089292031548145666

Progressive Web App

News on main OS:

Android

https://medium.com/@firt/google-play-store-now-open-for-progressive-web-apps-ec6f3c6ff3cc

Progressive Web App

News on main OS:

Android

Introducing a Trusted Web Activity for Android

Tuesday, February 5, 2019

A Trusted Web Activity (TWA) displays a full screen Chrome browser inside of an Android app with no browser UI. Although Android apps routinely include web content using a Chrome Custom Tab (CCT) or WebView, a TWA offers unique advantages when you need Chrome's performance and features in your app in full screen mode.

In this post I'll introduce you to TWAs, review suggested use cases and link to resources to get you started.

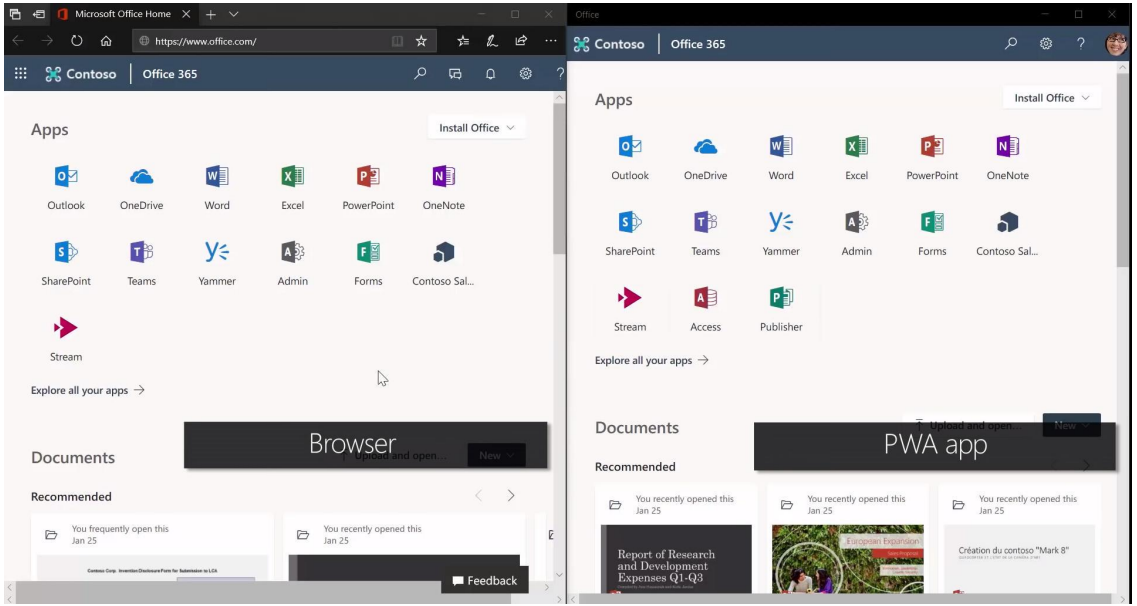
https://blog.chromium.org/2019/02/introducing-trusted-web-activity-for.html

Progressive Web App

News on main OS:

Windows

- Install from windows store
- Offline support
- Directly install from Edge
- Same API as UWP

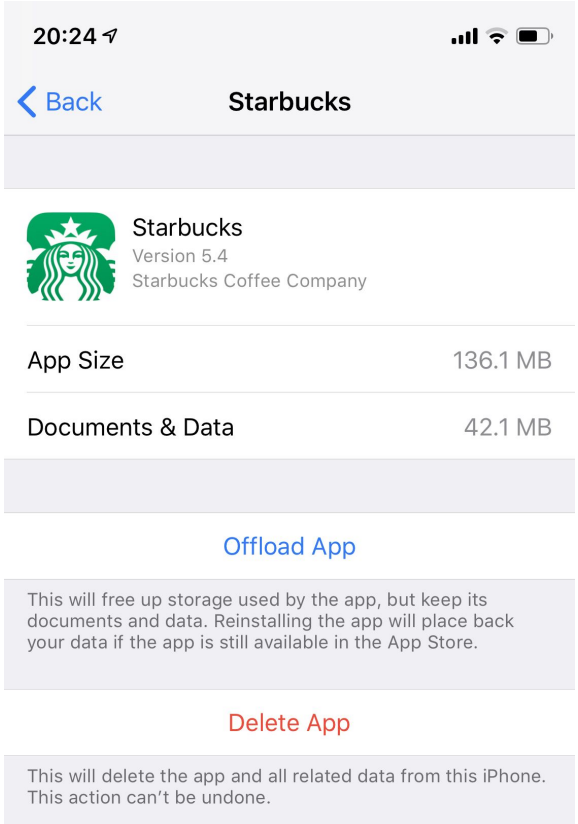


https://www.zdnet.com/article/windows-10s-new-free-office-app-windows-store-pwa-now-open-to-all/
https://www.youtube.com/watch?time_continue=366&v=VNpoqUNMrh8

Progressive Web App

Why use PWA

1. Size!!!



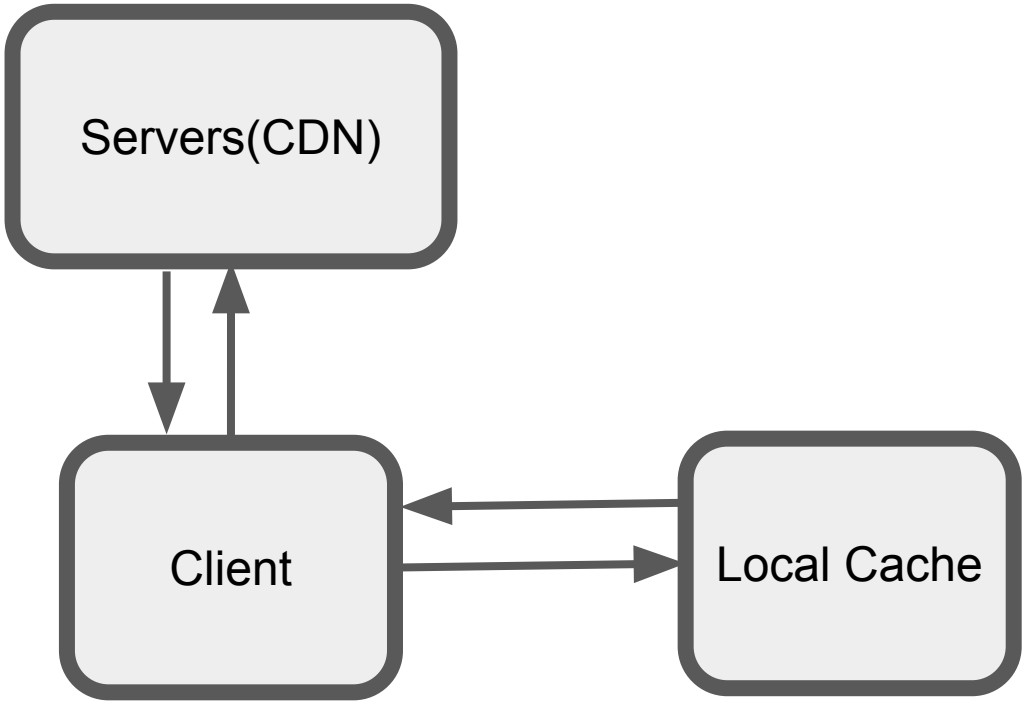
Progressive Web App

Why use PWA

1. Size!!!
2. Offline support: Service Worker
3. Native like app
4. Free of app store

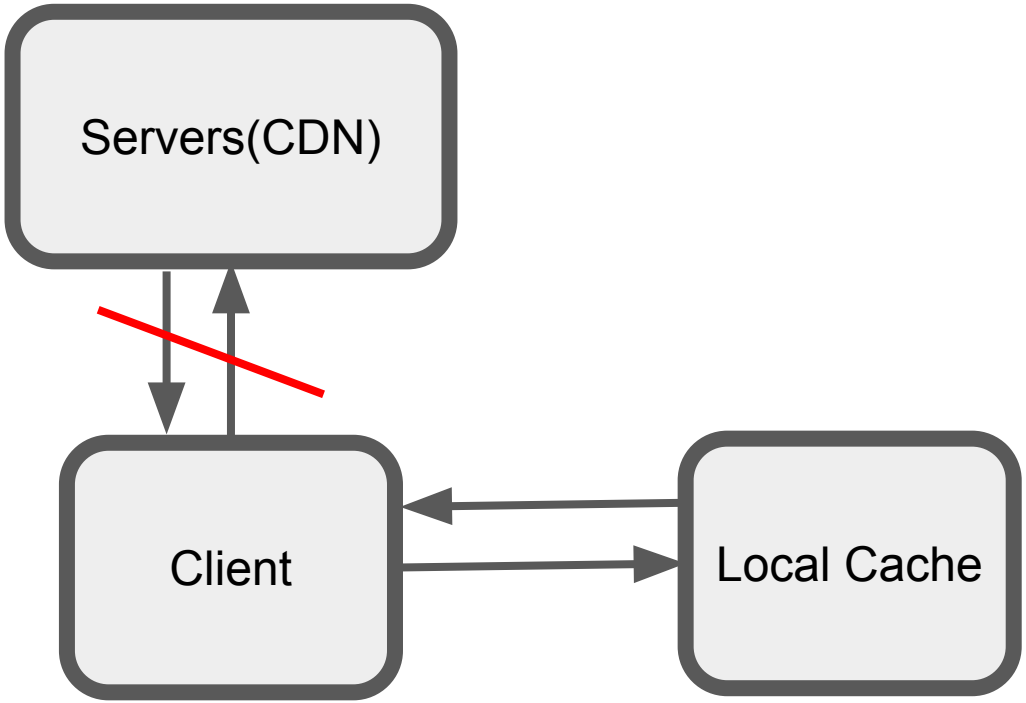
Progressive Web App

How PWA works offline



Progressive Web App

How PWA works offline



Progressive Web App

How PWA works offline

No internet

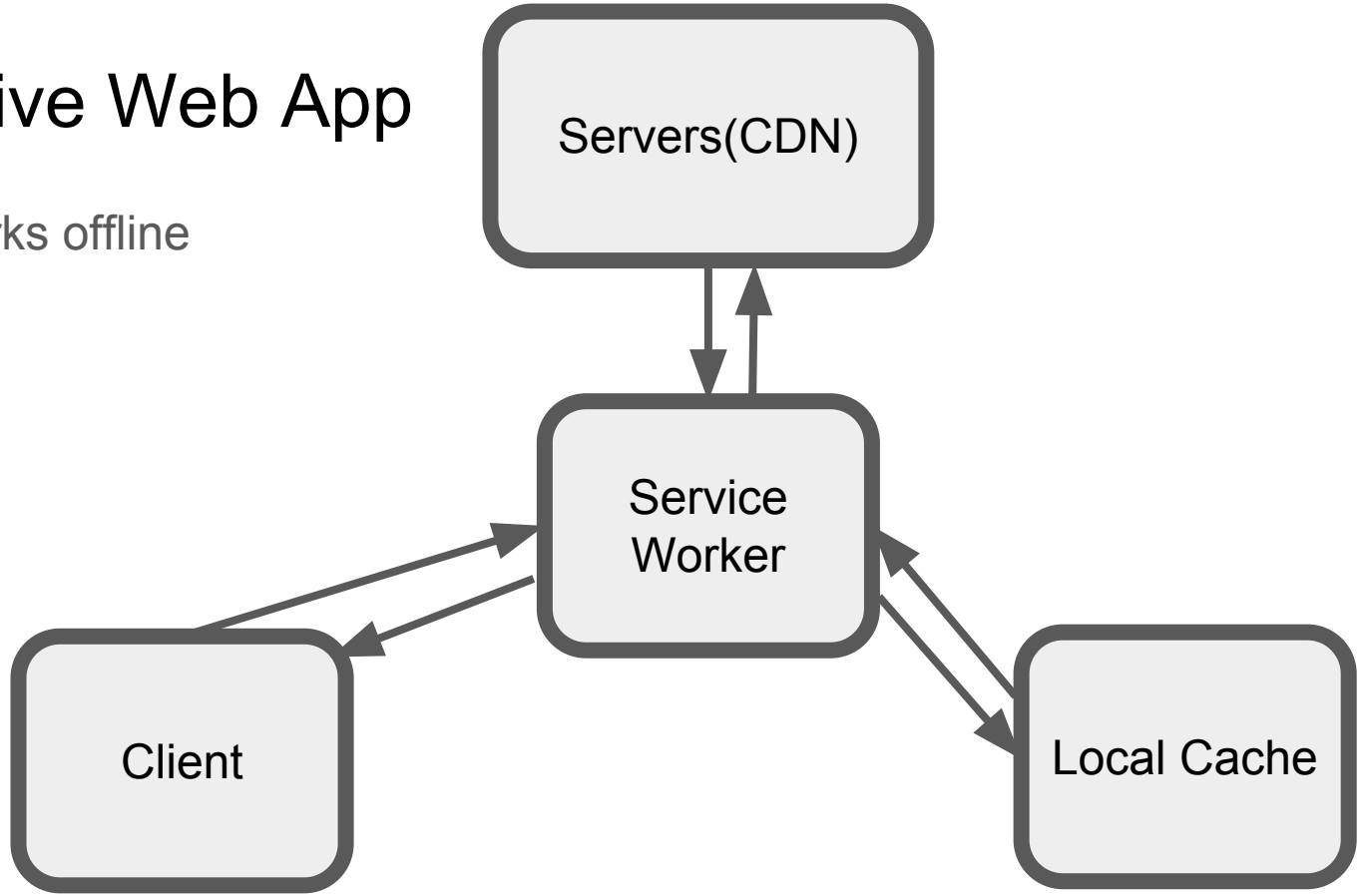
Try:

- Checking the network cables, modem, and router
- Reconnecting to Wi-Fi

ERR_INTERNET_DISCONNECTED

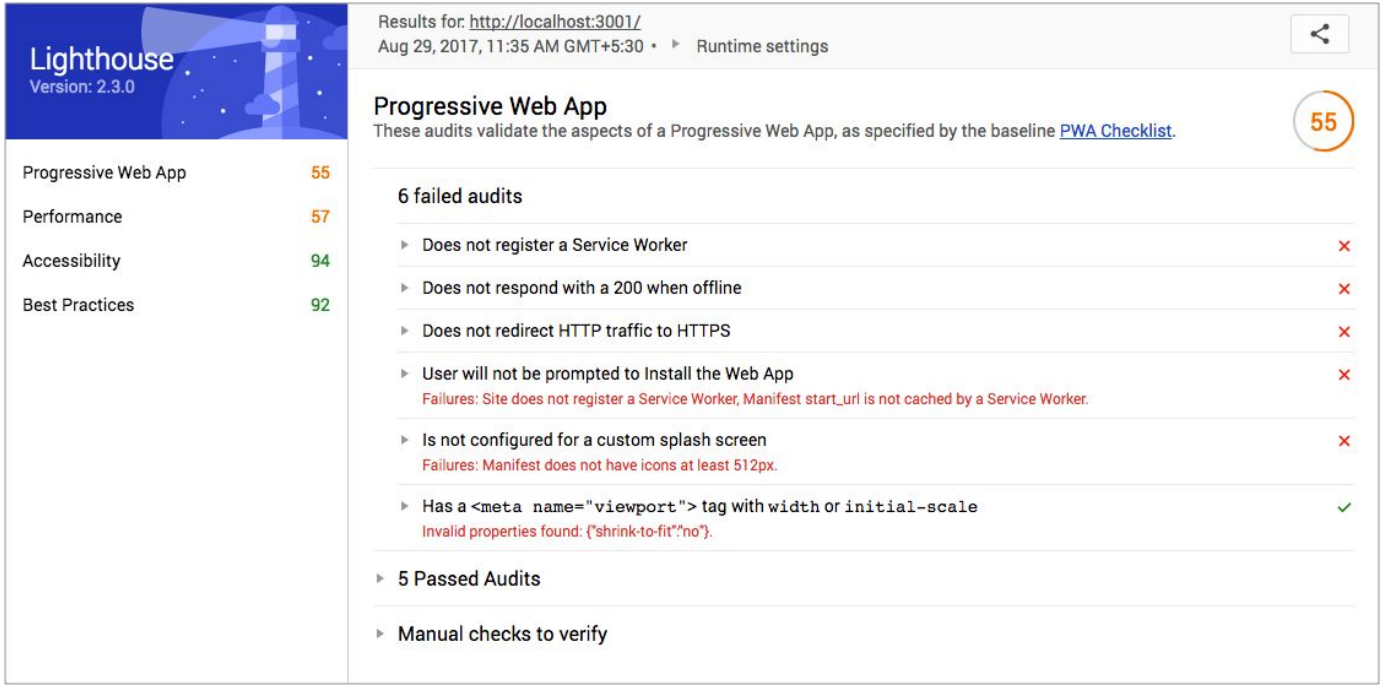
Progressive Web App

How PWA works offline



Progressive Web App

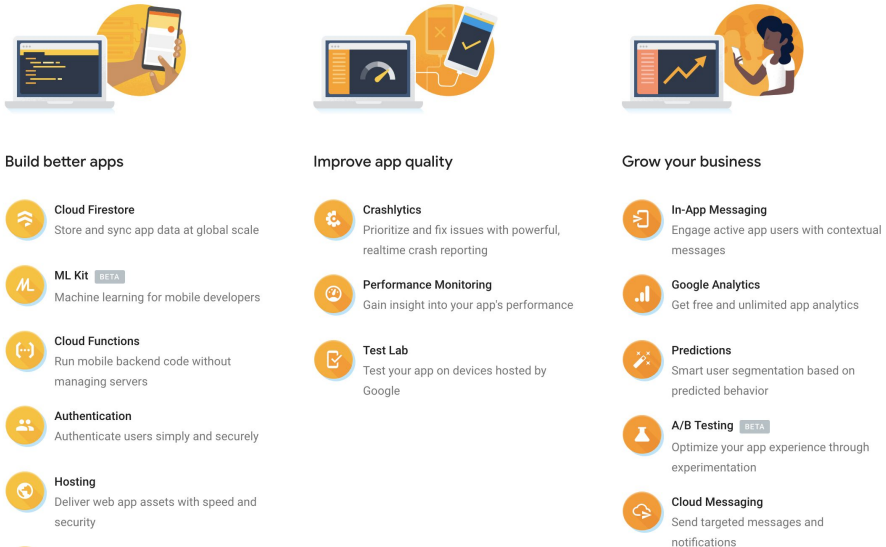
Use Lighthouse's checklist



Firebase

What’s Firebase

- Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014.
- Backend as a Service (BaaS)



Firebase

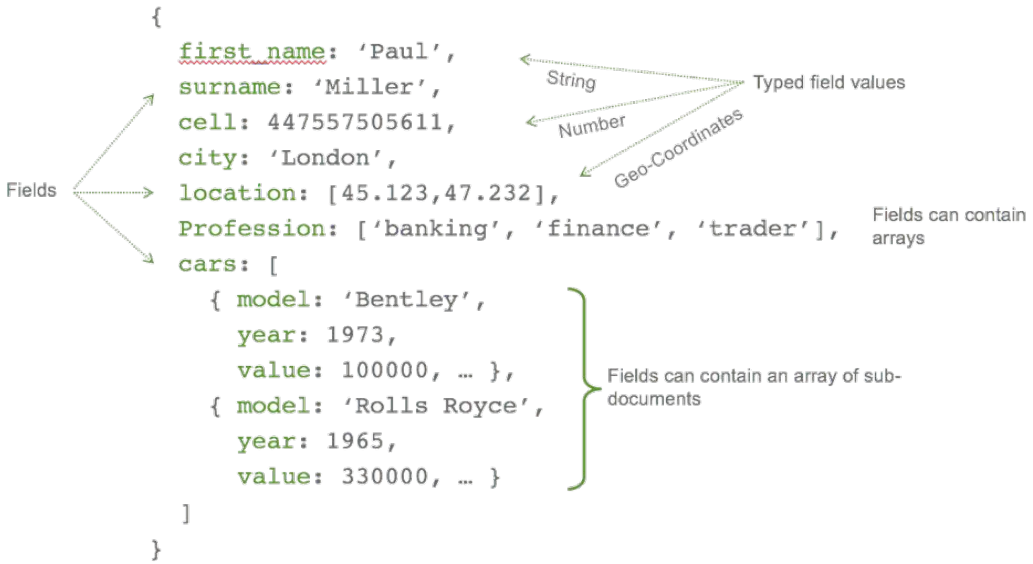
Database

- Firebase Realtime database
  - Cloud-host NoSql database
- Cloud FireStore
  - NoSql document based database

Firebase

Cloud FireStore, NoSql document based database on Google Cloud Platform

- Data Model
  - Document and collection

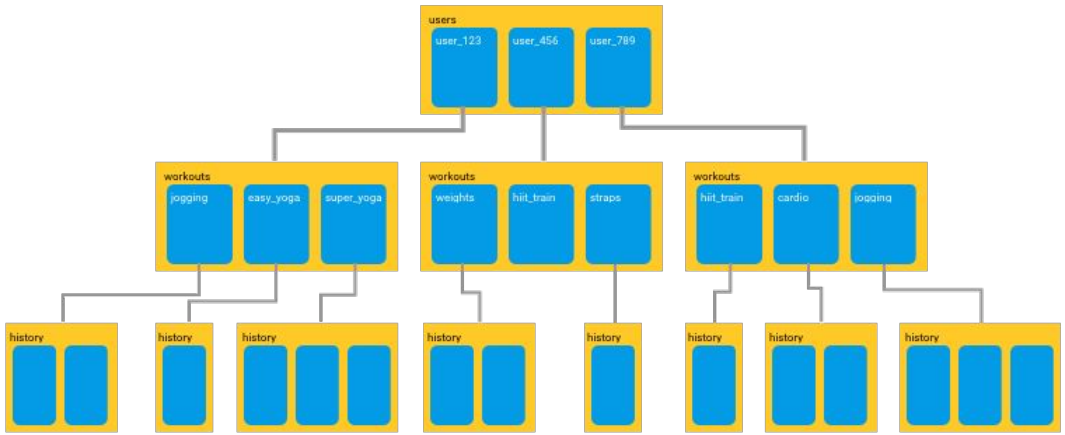


Tree-like JSON

Firebase

Cloud FireStore, NoSql document based database on Google Cloud Platform

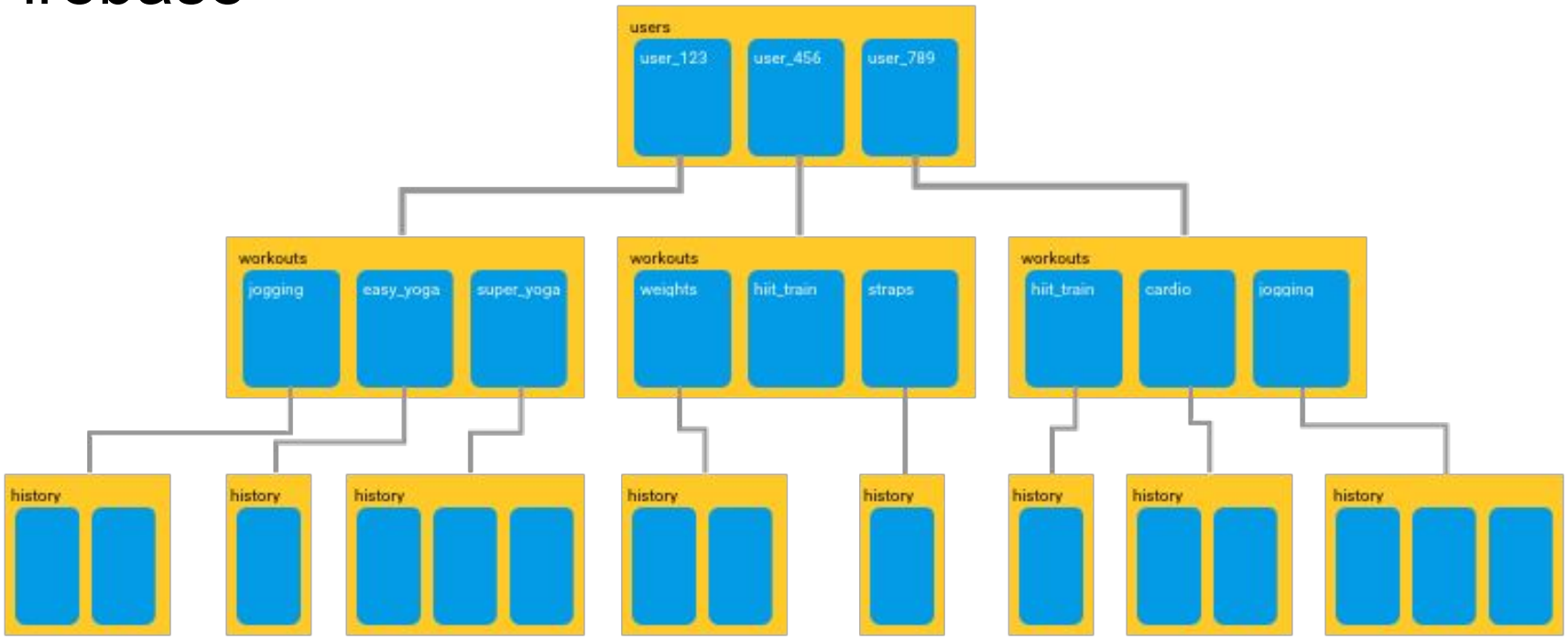
- Data Model
  - Document and collection



Document based Structure

https://proandroiddev.com/working-with-firestore-building-a-simple-database-model-79a5ce2692cb

Firebase



Document based Structure

https://proandroiddev.com/working-with-firestore-building-a-simple-database-model-79a5ce2692cb

Firebase - Cloud Firestore

Some constraints

- Collections can only contain document
- Document can not contain another document
- Only document can have fields

Firebase - Cloud Firestore

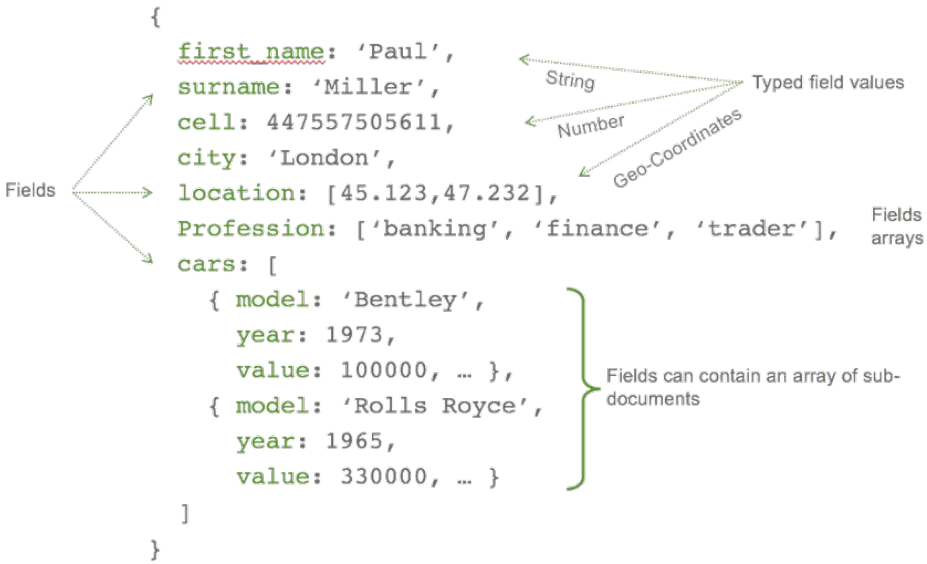
Read Data

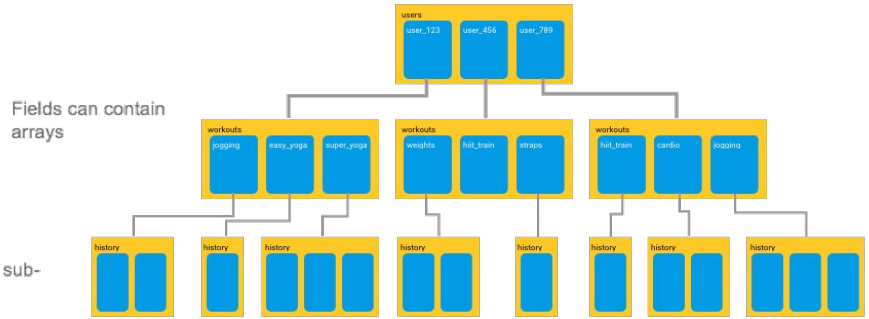
- firestore.collection(...).document(...).collection(...)
- Or
- “users/user_123/workouts/jogging/history/...”

Firebase - Cloud Firestore

How Query works in Cloud Firestore

- Queries are shallow by default





Firebase - Cloud Firestore

How Query works in Cloud Firestore

- Query performance is proportional to the size of your result set, not your data set.
- Why? Indexing every field by default, so binary search is possible

Task Division and Timeline

Task Division & Timeline

Before March	To Apr. 15th	Apr. 15th - Apr. 28th
System Design	Implementation	Final Integration

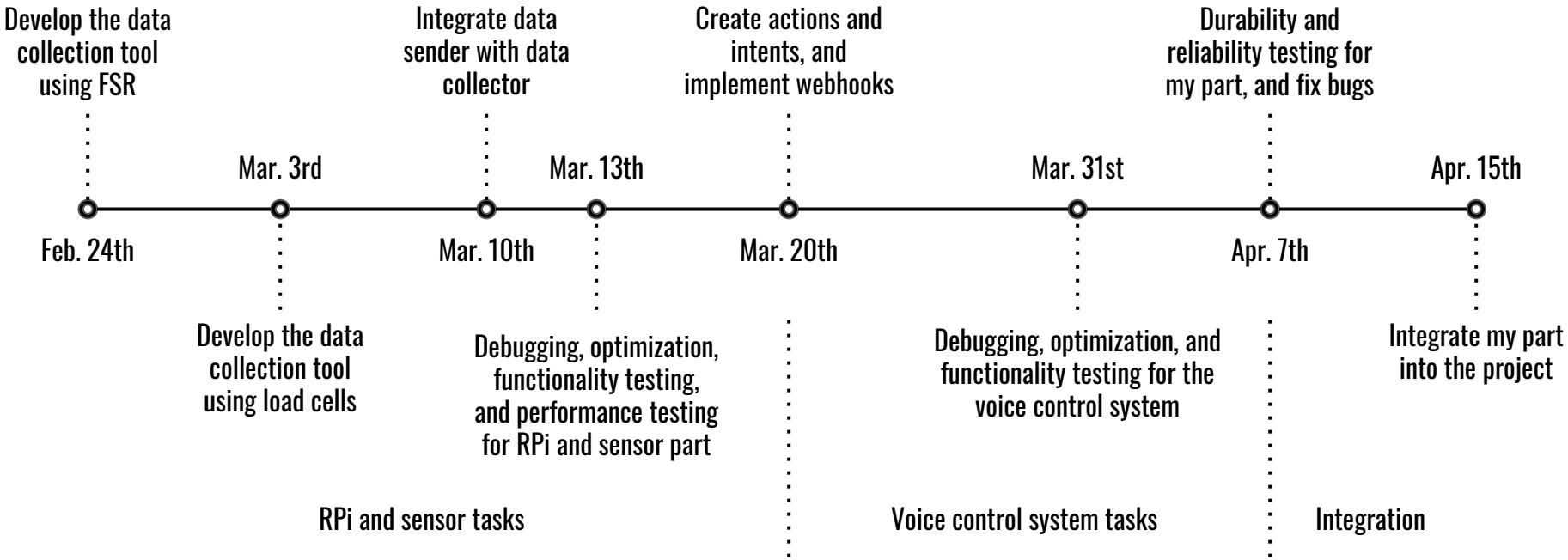
Yang He: Sensor data collection and Voice Control System

Cheng Wang: TaskLogger Website

Xu Zhang: Sitting website

Timeline

Yang He - Sensor data collection and Voice Control System

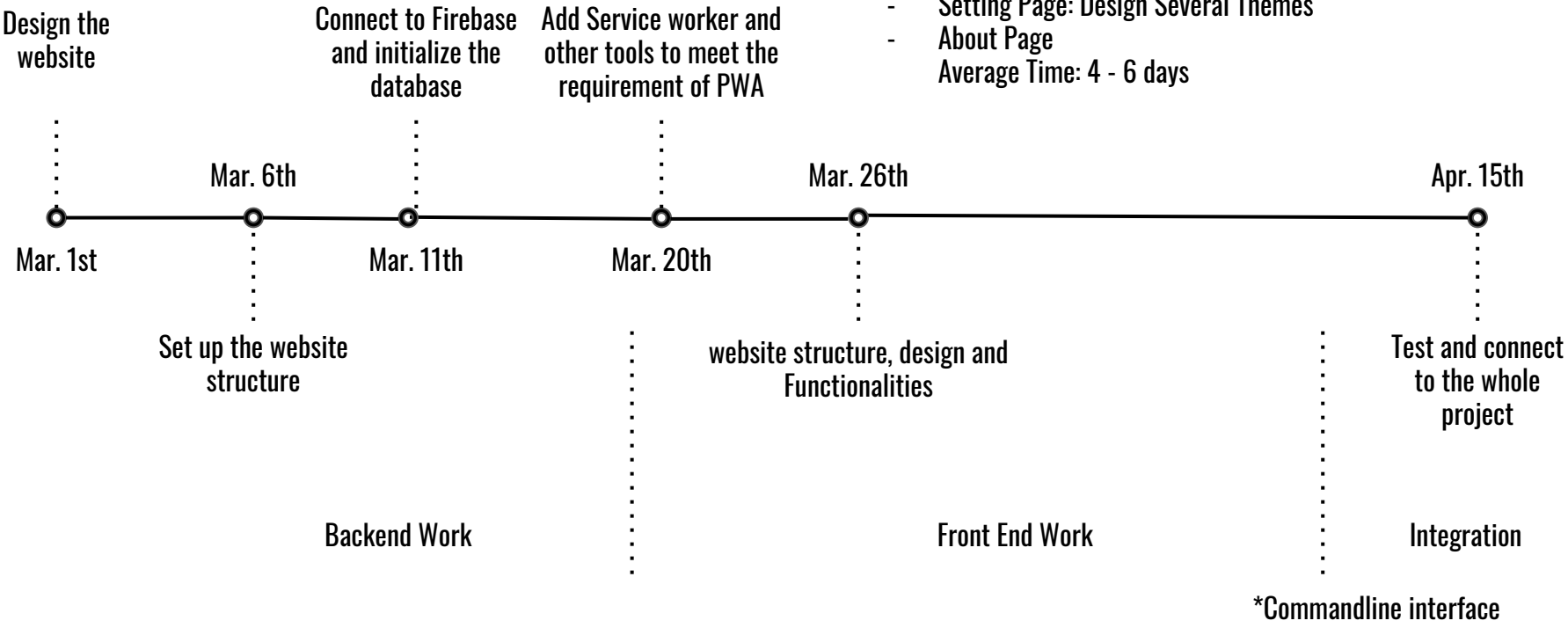


Timeline

Cheng Wang - TaskLogger Website

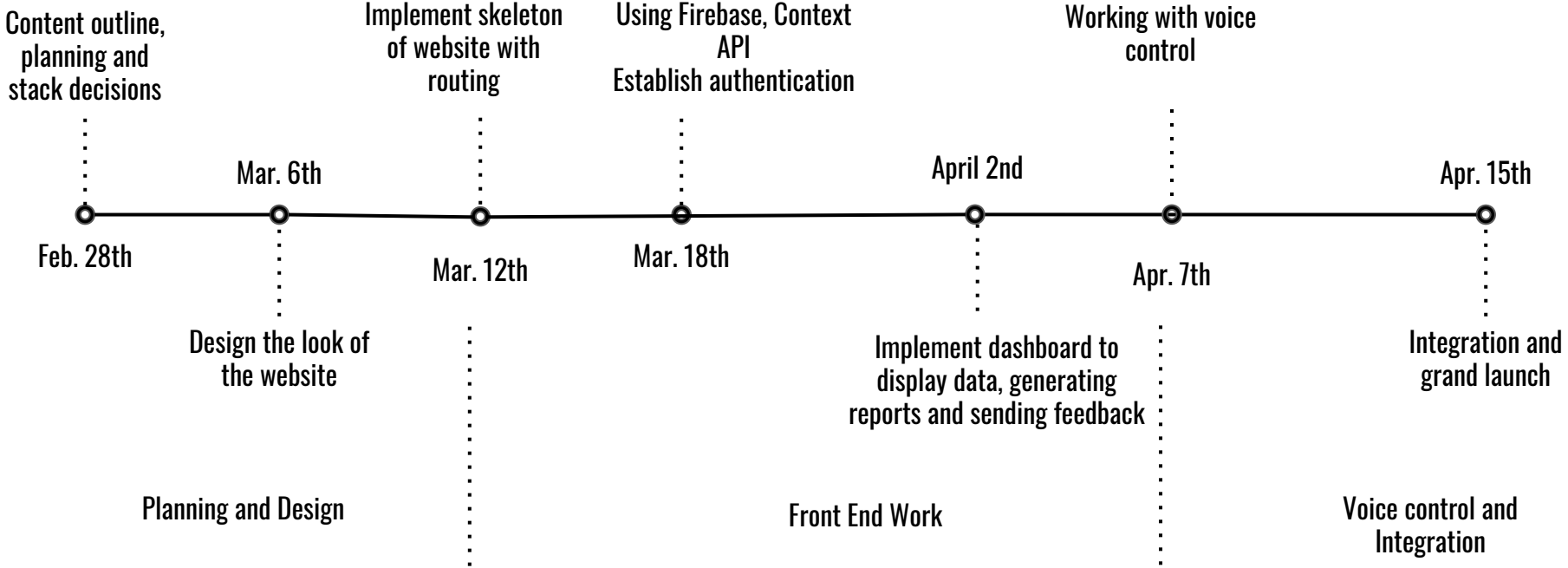
Front-end Work, structure and design
- Everyday page: interaction
- Report Page: generate report
- Export Page: connect to Google API
- Setting Page: Design Several Themes
- About Page

Average Time: 4 - 6 days



Timeline

Xu Zhang - Sitting website



Q & A