# 2. Part 1: IFTTT clone /45

**Keywords**: services, json, flows, stringify, IFTTT, JSON configuration file, dictionary of services, array of flows, API key

A **service** is a named dictionary which contains the program to be run, along with any needed parameters and configuration information.

A **flow** is a named array of services to be called in sequence, with any output from one being inserted as the input to the next service.

Specs:

1. command line input for .json config file, default is ifttt.json
   1. abort if invalid JSON or missing mandatory sections, return the line the error was found on.
   2. Dictionary of services and an array of flows

## 2.1. Identifies config file/alternative – 2

Ifttt.json is the config file, if not, then any .json (or .ini?) file passed into the command line?

API key saved into config file

Geolocation API beforehand where-am-i.co

use metric

use json formatter to test your config file

## 2.2. Reads in json – 3

## 2.3. Exits on invalid config format – 3

## 2.4. Run flows – 8

Status code of 0 = stop

## 2.5. Runs services – 6

## 2.6. Inputs from one service passed to the next – 6

## 2.6. Runs every minute - 2

The program should commence cycling through all the flows at the start of every minute. However if a flow runs overtime this rule won't apply.

## 2.7. Services – 6

## 2.8. special symbol $$ substitution – 6

## 2.9. Single flow stops on fail result of a service – 3

# 3. Part 2: More complicated IFTTT clone /45

## 3.1. Loads config file on changes – 5

## 3.2. Reloads service description on changes – 5

## 3.3. Configuration file formatted validated – 15

## 3.4. Services accept/emit json format previous services recoded for new format – 10

## 3.5. Program ensures data passed is formatted as specified stops the flow if in error – 10