Simple Financial Planner

## Objective

The Simple Financial Planner application allows users to determine their retirement income based on a variety of assumptions, such as funds available today, asset allocation strategy, inflation rate, retirement age, etc.

## Versions

Version 1 is a simple demo app which will not implement any of the concepts expected in a production app: security, user & user settings, nor data persistence (tbd).

The main entity used in Version 1 is a **Financial Plan “*FinPlan”:*** a set of assumptions about the user’s financial state and goals - and associated results. To the extent the client (mobile app) can store the ID (*FinPlanID*) of one or more FinPlans the user will be able to view and edit FinPlans created in previous sessions. *FinPlanID* is, in Version 1, the only way to access a Financial Plan

## Features

* A FinPlan contains phases, with different investment strategies: e.g. contribution/withdrawal, asset allocation
* Within a phase, the user specifies asset allocation percentage among: Stocks, Bonds, T-Bills and Cash. The user also sets NetContribution: Contribution/Withdrawal if positive/negative number. User can also select whether the withdrawal amount will be computed by Monte-Carlo simulation. More than one phase can set this flag, but the amount will be common for these phases. See Usage Example below
* The Simple Financial Planner has the historical statistical variables for the 4 types of assets
* Add Monte-Carlo simulation
* Add historical return rates for various investment styles to feed the Monte Carlo simulation

# API / URL Routes

The functionality provided by the FinPlan app is:

* Create: a new scenario
* View an existing scenario
* Edit an existing scenario
* Delete an existing scenario
* Run / compute a given scenario

The URL Routes are:

| HTTP Method | URI | Action |
| --- | --- | --- |
| GET | <http://[>hostname]/finplan/api/v2.0/finplan | Retrieve list of FinPlans |
| POST | <http://[>hostname]/finplan/api/v2.0/finplan | Create a new FinPlan |
| GET | <http://[>hostname]/finplan/api/v2.0/finplan/[finplan\_id] | View a FinPlan |
| PUT | <http://[>hostname]/finplan/api/v2.0/finplan/[finplan\_id] | Update a FinPlan |
| DELETE | <http://[>hostname]/finplan/api/v2.0/finplan/[finplan\_id] | Delete a FinPlan |
| GET | <http://[>hostname]/finplan/api/v2.0/run/[finplan\_id/compute | Compute a FinPlan |

# FinPlan

## Create

A FinPlan CREATE operation must have the following parameters:

* Title: name - String type
* Description - String type. (e.g. to contrast it to other FinPlans)
* UserName - String type - Name of the user
* Email - String type - User Email

The Response will contain the above parameters PLUS:

* FinPlan\_ID: ID - this auto\_generated by the server, and is the parameter to be used for any subsequent access to the
* HasResult - Boolean - False by default - indicates whether the scenario was run and includes the results. Any Update operation will set HasResult to False

## General Parameters

## Phase Parameters

| Phase Parameter | Type | Description | Default |
| --- | --- | --- | --- |
| Name | String | Name | Required |
| StartAge | Int | Starting age for this phase | Required |
| EndAge | Int | End age for this phase | Required |
| NetContribution | Int | Yearly contribution/withdrawal if positive/negative | Value is ignored if ToCompute flag is set to True |
| ToCompute | Flag | if True: NetContribution amount will be computed by Monte-Carlo simulations  if False: NetContribution value will be used for computations | 0 |
| Portfolio | Dict | Dict containing asset allocation percentages  Example:  'Portfolio': {"Stocks": 80.0, "Bonds": 20.0, "T-Bills": 0.0, "Cash": 0.0} | Required |

# Usage

**fin plan -c confidence\_percentage -r run\_count\_for\_monte\_carlo -N -D**

| General Parameter | Type | Description | Default |
| --- | --- | --- | --- |
| FinPlan\_ID | UUID String | ID | Auto-generated by server |
| Title | String | Name | Required |
| Description | String | Description | Optional |
| UserName | String | Name of the user | Required |
| Email | String | User Email | Required |
| HasResult | Boolean | Results have been computed | 0 |
| AgeToday | Int | User’s age at start of plan | Required |
| AgeFinal | Int | Age at end of retirement | Optional |
| StartingAmount | Int | Retirement funds available today | Required |
| InflationRate | Float | Average inflation rate across the whole period  Expressed in % | 2.0 |

| Argument | Description | Default |
| --- | --- | --- |
| confidence\_percentage | Confidence percentage in Monte-Carlo simulations to determine withdrawal ammount | 85% |
| run\_count\_for\_monte\_carlo | Number of Monte-Carlo simulations run | 10,000 |
| -N | If present, the program starts with default list of FinPlanScenarios, rather than reading from persistence store | Reads from persistence store |
| -D | If present - Debug is True | 0 |

All arguments are optional

# Future

* Replace current file by database as persistence store
  + With file, the whole file is re-written, each time one of the FinPlans is created/modified/deleted
* Add yearly portfolio rebalancing
* Add createdDate, lastModifiedDate
* Add automatic portfolio builder and a Risk-Based Portfolio allocation tool - see “Risk-Based-Portfolio) attached. E.g. Risk = 8 during working phase, Risk = 5 at start of retirement, Risk = 1 last 10 years of retirement
* Add gender and set default end retirement age based on statistics
* Add users, user profile (some of the data in scenario will be part of user profile - e.g. email, name, age, …)
* Add security: e.g. only the owner of a FinPlan can view/modify it
* Add more asset types and the ability for user to create their own custom asset type
* Add tax rate
* Add web app
* Add Sensitivity Analysis - see: <http://www.flexibleretirementplanner.com/wp/documentation/sensitivity-analysis/>