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, investment return rate, 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

# 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/v1.0/finplan | Retrieve list of FinPlans |
| POST | <http://[>hostname]/finplan/api/v1.0/finplan | Create a new FinPlan |
| GET | <http://[>hostname]/finplan/api/v1.0/finplan/[finplan\_id] | View a FinPlan |
| PUT | <http://[>hostname]/finplan/api/v1.0/finplan/[finplan\_id] | Update a FinPlan |
| DELETE | <http://[>hostname]/finplan/api/v1.0/finplan/[finplan\_id] | Delete a FinPlan |
| POST | <http://[>hostname]/finplan/api/v1.0/run/[finplan\_id] | 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 cause the server to set HasResult to False

## Parameters

| Parameter | Type | Description | Default |
| --- | --- | --- | --- |
| FinPlan\_ID | UUID String | ID | Auto-generated by server |
| Title | String | Name | Required |
| Description | String | Description | Required |
| 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 | 30 |
| AgeRetirement | Int | User’s age at retirement | 65 |
| AgeEnd | Int | Age at end of retirement | 90 |
| StartingAmount | Int | Retirement funds available today | 200,000 |
| PreContribution | Int | Yearly contribution prior to retirement | 17,500 |
| Withdrawal | Int | Year withdrawal during retirement | none |
| PreReturnRate | Float | Return rate on investment prior to retirement  Expressed in % | 7.0 |
| RetirementReturnRate | Float | Return rate on investment during retirement  Expressed in % | 5.0 |
| InflationRate | Float | Average inflation rate across the whole period  Expressed in % | 2.0 |

# Future

* Add Phases within a FinPlan - each phase has different assumptions (e.g. contributions/withdrawals, investment return rate / portfolio, etc)
* Add Portfolio: e.g. Stocks/Bonds/Cash - each with its own investment return rate
* Add Monte-Carlo simulation
* Add historical return rates for various investment styles to feed the Monte Carlo simulation
* 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