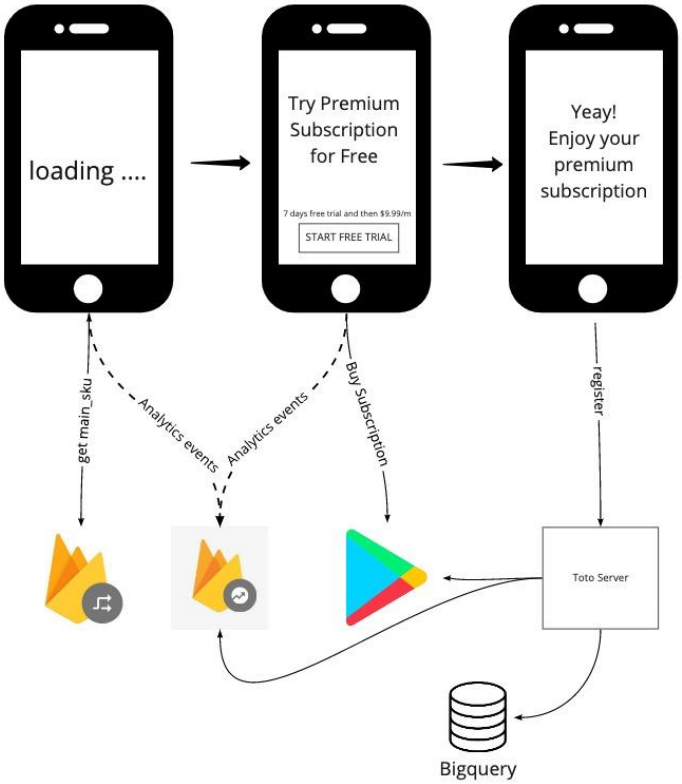
Toto Server Config API - Test Project

Summary

The Toto server is our “central brain” for executing and measuring all the in-app purchases made throughout our apps. Once we onboard a new app we acquired, we incorporate our SDK which communicates with the Toto server (among other things).

The purpose of this task is to replace the use of Firebase Remote Config and to expose an API directly on Toto server for distributing our premium offering configuration to all our apps.

Detailed Overview

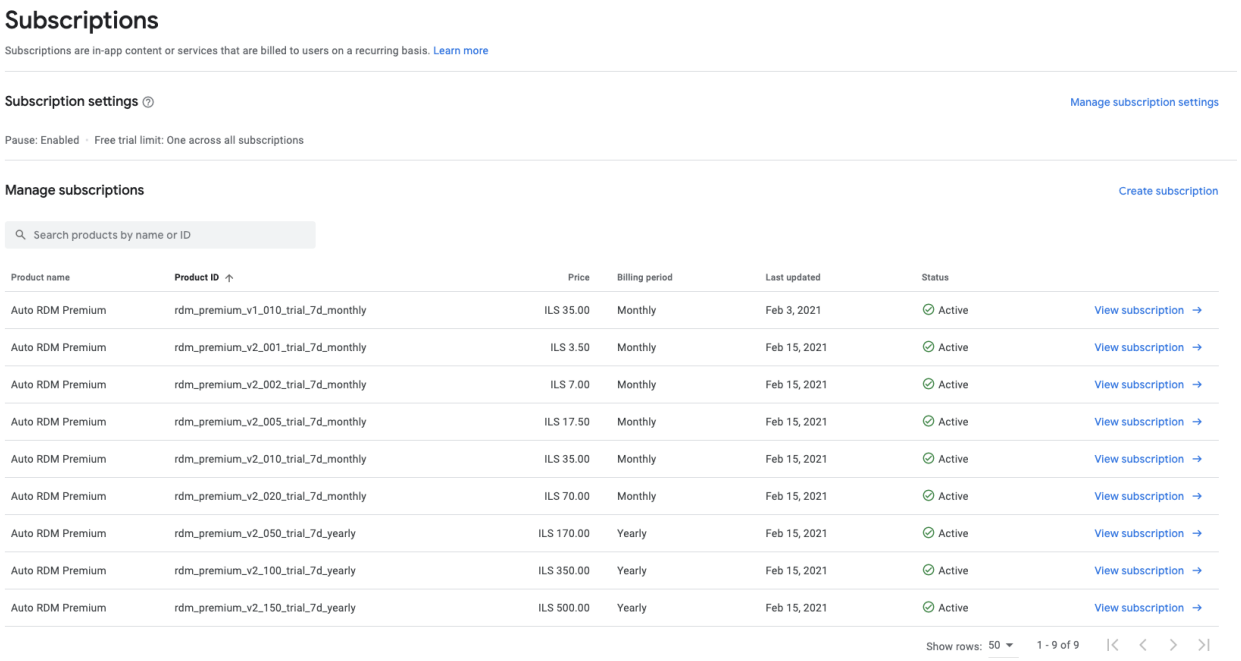


Currently Toto is a somewhat “hacked” solution for managing and measuring our premium offerings experiments. It consists of the following parts:

Pricing offering configuration

This is the component that is in charge of configuring the subscription offer we want to present to the user. We currently use Firebase Remote Config for this purpose (link).
Following are the steps we make when we conduct a pricing test within one of the apps:

1. We create a few different subscription offerings on Google Play (for instance: $1/m with 7days trial, $5/m, $100/y, …) - This is currently done manually
   This is how the subscriptions are managed within Google Play:



Subscriptions

Subscriptions are in-app content or services that are billed to users on a recurring basis. Learn more

Subscription settings Manage subscription settings

Pause: Enabled · Free trial limit: One across all subscriptions

Manage subscriptions Create subscription

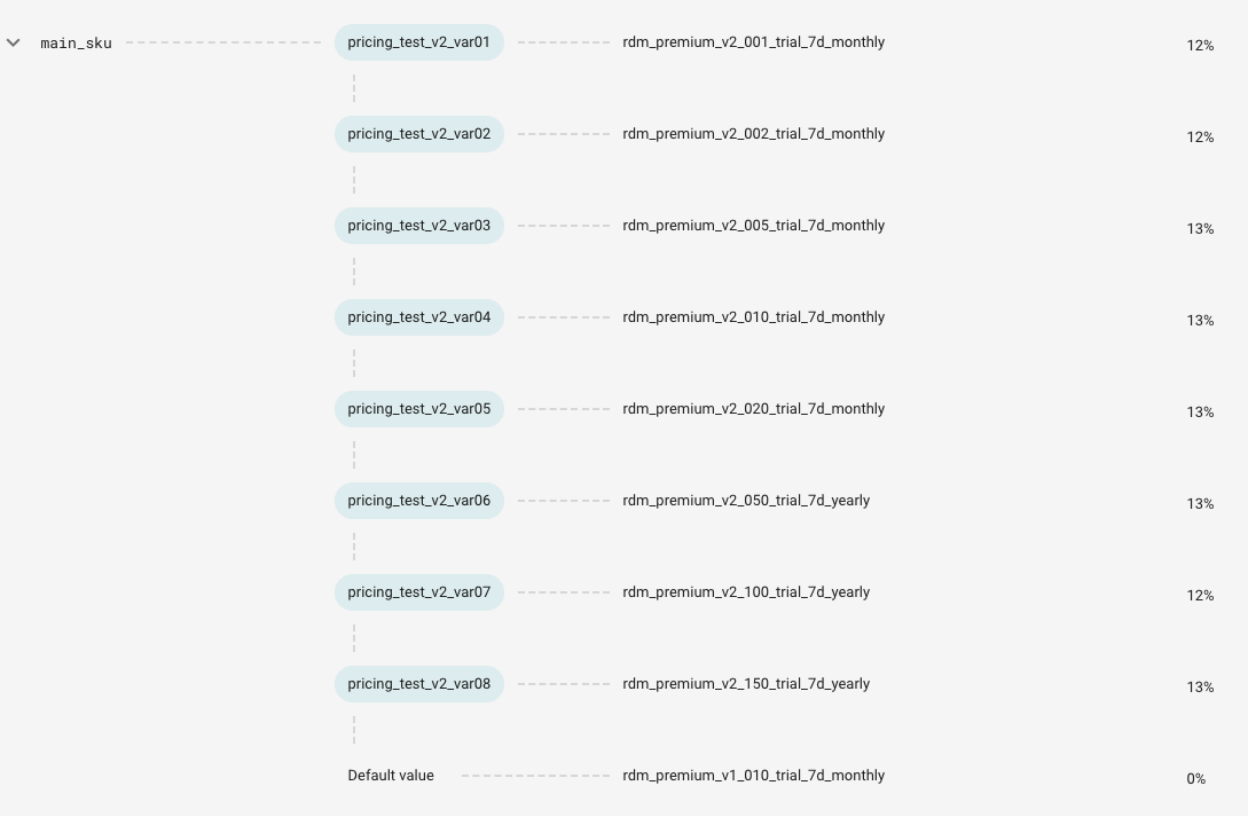
Search products by name or ID

Product name	Product ID	Price	Billing period	Last updated	Status	
Auto RDM Premium	rdm_premium_v1_010_trial_7d_monthly	ILS 35.00	Monthly	Feb 3, 2021	Active	View subscription
Auto RDM Premium	rdm_premium_v2_001_trial_7d_monthly	ILS 3.50	Monthly	Feb 15, 2021	Active	View subscription
Auto RDM Premium	rdm_premium_v2_002_trial_7d_monthly	ILS 7.00	Monthly	Feb 15, 2021	Active	View subscription
Auto RDM Premium	rdm_premium_v2_005_trial_7d_monthly	ILS 17.50	Monthly	Feb 15, 2021	Active	View subscription
Auto RDM Premium	rdm_premium_v2_010_trial_7d_monthly	ILS 35.00	Monthly	Feb 15, 2021	Active	View subscription
Auto RDM Premium	rdm_premium_v2_020_trial_7d_monthly	ILS 70.00	Monthly	Feb 15, 2021	Active	View subscription
Auto RDM Premium	rdm_premium_v2_050_trial_7d_yearly	ILS 170.00	Yearly	Feb 15, 2021	Active	View subscription
Auto RDM Premium	rdm_premium_v2_100_trial_7d_yearly	ILS 350.00	Yearly	Feb 15, 2021	Active	View subscription
Auto RDM Premium	rdm_premium_v2_150_trial_7d_yearly	ILS 500.00	Yearly	Feb 15, 2021	Active	View subscription

Show rows: 50 1 - 9 of 9

2. We configure the test on Remote Config to randomly distribute the offerings between the users - also done manually

This is how it looks on Firebase Remote Config:



Storing the monetization related data

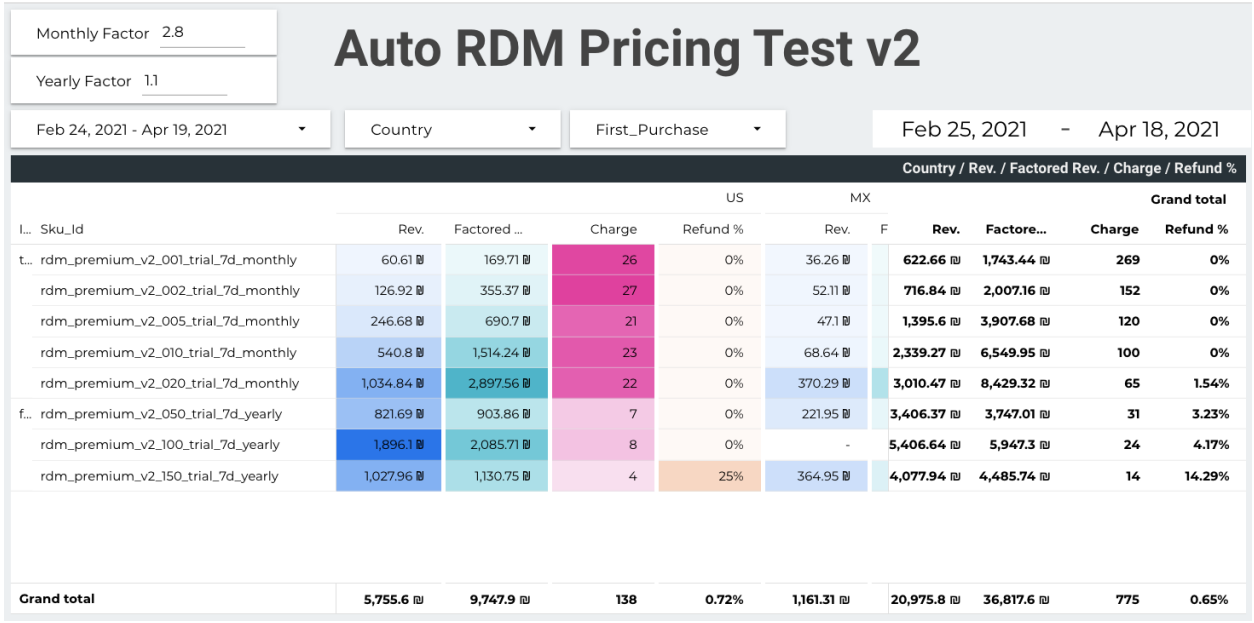
In order to analyze the different pricing tests and pick the winning variant, Toto stores monetization related data with the following components:

1. Exposing a register API which is called from the app itself after successful payment. This helps us attribute the subscription payment with the source traffic (organic user, paid campaign, and so on..)
2. We have some CRON jobs which pull the data from Firebase Analytics and Google Play and store them in Bigquery.

Pricing test results examine

1. We examine the results of the pricing test via a Data Studio report which visualizes the data which resides in Bigquery. This is how currently look at the results of the pricing test

manually on a Data Studio report:



Monthly Factor 2.8

Yearly Factor 1.1

Auto RDM Pricing Test v2

Feb 24, 2021 - Apr 19, 2021 | Country | First_Purchase | Feb 25, 2021 - Apr 18, 2021

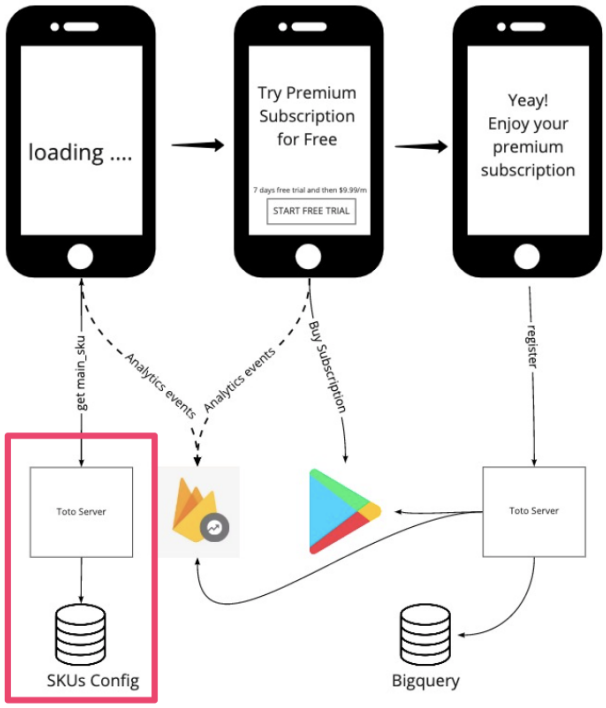
Country / Rev. / Factored Rev. / Charge / Refund %

		US				MX		Grand total			
I...	Sku_Id	Rev.	Factored ...	Charge	Refund %	Rev.	F	Rev.	Factore...	Charge	Refund %
t...	rdm_premium_v2_001_trial_7d_monthly	60.61 ₪	169.71 ₪	26	0%	36.26 ₪		622.66 ₪	1,743.44 ₪	269	0%
	rdm_premium_v2_002_trial_7d_monthly	126.92 ₪	355.37 ₪	27	0%	52.11 ₪		716.84 ₪	2,007.16 ₪	152	0%
	rdm_premium_v2_005_trial_7d_monthly	246.68 ₪	690.7 ₪	21	0%	47.1 ₪		1,395.6 ₪	3,907.68 ₪	120	0%
	rdm_premium_v2_010_trial_7d_monthly	540.8 ₪	1,514.24 ₪	23	0%	68.64 ₪		2,339.27 ₪	6,549.95 ₪	100	0%
	rdm_premium_v2_020_trial_7d_monthly	1,034.84 ₪	2,897.56 ₪	22	0%	370.29 ₪		3,010.47 ₪	8,429.32 ₪	65	1.54%
f...	rdm_premium_v2_050_trial_7d_yearly	821.69 ₪	903.86 ₪	7	0%	221.95 ₪		3,406.37 ₪	3,747.01 ₪	31	3.23%
	rdm_premium_v2_100_trial_7d_yearly	1,896.1 ₪	2,085.71 ₪	8	0%	-		5,406.64 ₪	5,947.3 ₪	24	4.17%
	rdm_premium_v2_150_trial_7d_yearly	1,027.96 ₪	1,130.75 ₪	4	25%	364.95 ₪		4,077.94 ₪	4,485.74 ₪	14	14.29%
Grand total		**5,755.6 ₪**	**9,747.9 ₪**	**138**	**0.72%**	**1,161.31 ₪**		**20,975.8 ₪**	**36,817.6 ₪**	**775**	**0.65%**

2. We create a subsequent test or pick the winner offering via Remote Config.

Task Requirements

In this task we want to replace the current use of Firebase Remote Config to set the SKU for the user with our own developed API. Something like this:



For the sake of simplicity, for this first task we will focus only on the main subscription sku and the country the user is coming from.

API Request Parameters

Name	Example	Explanation
`package`	`com.softinit.iquitos.mainapp`	`This is the identifier of the calling app`

API Response Parameters

Name	Example	Explanation
`main_sku`	`rdm_premium_v2_002_trial_7d_monthly`	`This is the subscription offer id (sku)`

Configuration Table

The configuration table will have the following columns:

- `package` - the app package identifier
- `country_code` - the 2 letters code identifying the user (based on the IP the user originates from). “ZZ” will be used as a default rule if the specific country the user is coming from doesn’t have a specific rule.
- `percentile_min` - In order to support randomly distributed tests. We randomly choose a number between 1 and 100 for the incoming API call and pick the configurations that are greater than this percentile
- `percentile_max` - the same as min but just for less than or equal to
- `main_sku` - the SKU to return to the user

These are the values we could start working with:

package	country_code	percentile_min	percentile_max	main_sku
com.softinit.iquitos.mainapp	US	0	25	rdm_premium_v3_020_trial_7d_monthly
com.softinit.iquitos.mainapp	US	25	50	rdm_premium_v3_030_trial_7d_monthly
com.softinit.iquitos.mainapp	US	50	75	rdm_premium_v3_100_trial_7d_yearly
com.softinit.iquitos.mainapp	US	75	100	rdm_premium_v3_150_trial_7d_yearly
com.softinit.iquitos.mainapp	ZZ	0	100	rdm_premium_v3_050_trial_7d_yearly

Things to Consider

- Low latency
  Since this API is called while the user is seeing the loading screen, it needs to be FAST. We should aim for 95% of the requests to finish within 200ms.

- Globally Spread
  Our users are really globally spread and the API should be low latency for all of them. USA is of course our top priority.

- The scale
  The throughput we are aiming for by the end of the year is somewhere between 2-5 million calls a day (~50/s)

- User country
  We need to identify the user country based on the IP address the user is coming from. BTW Google App Engine has a built-in solution where the country code is passed inside as an HTTP header.

Deliverables

1. An architecture plan for the config API solution including the deployment topology.
2. Build and deploy a mockup API which we will run under some global load test.