Lewis Daly
CTO of vesselstech.com | Digital Tech for Development
Draft

How I'm finding the price to set my new product

At Vessels, we are building a super cool SMS bot for data collection. SMS is a tool used all accross the world for data collection and research, and we think we can combine that popularity and low-tech solution with the modern trend for chat bots. You can find out more here:

SMS Data collection Bots

Turn real conversations into real data with smscollect. smscollect is a chatbot plugin for twilio, nexmo and many more…

smscollect.xyz

We've made a decent headway into developing our protoype, and we already have our first customer (ourselves), but if we really want to spin this off into its own product, we need to figure out how much it will cost, and how many customers we need to make it sustainable and scaleable.

Let's start by dividing our costs across a few differerent categories:

- - **Business Costs:** Costs that we incurr by just keeping the lights on, such as database, server, network etc.

Per Customer costs: Costs incurred per user of the system, such as SMS gateways, and a portion of data centre based costs. Also the cost of onboarding and setting up new

Development, Support & Marketing: Investments in the product or brand that will increase the number of sales we make, and the number of customers we retain.

Business Costs

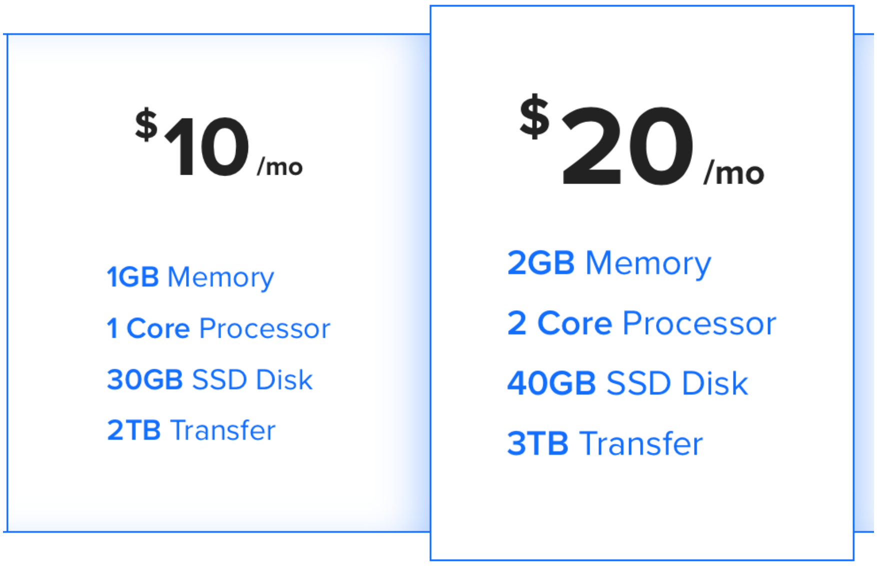
In the absence of any real users (or even a complete system), we will have to make up these costs based on previous experience. The SMS bot is managed as a collection of (currently) 2 docker containers, a conversation service and redis. We will likely expand this to 4 or 5, including a SMS bot management website for customers to manage their bot, a database, and possibly a NLU service (if we decide to move away from Wit.ai at this stage).

None of these services are all that CPU intensive, they will just be IO and Memory bound (with the exception of training the NLU service). Examining these 2 existing containers on my local machine at rest sees that they are using ~45MB of RAM. Sending a request to the sms-bot service seems to increase the RAM to ~47MB total. So, assuming (and this is a big assumption) that with adding a few more containers to this setup of at least 50MB at rest each, we get to a total of ~300MB to keep the lights on, with 0 users.

Making the assumption that each request costs 2MB for each service, we can predict that each request requires 5 * 2MB = 10MB across the whole system. At 1 request per second, (assuming it takes about 30 seconds for the whole system to release this memory), in 30 seconds we would be using 300MB, so 600MB total.

But how many requests could we expect per second? Well lets take our first customer, MyWell as an example. With MyWell, our peak load is only when well readings are recorded once a week. If the system is being used to track 1000 wells, and each recording takes 2 messages, we have to handle 2000 requests in the space of a week, it averages to about ~0.0033 requests/second. Not exactly a challenge. This blatantly overlooks the fact that well readings are not likely to be recorded at night or on weekends—in fact I believe monday is the day that most well readings are recorded for MyWell. Even if all 2000 requests were made within an 8 hour period, this works out to be: 2000 requests/8 hours/60 minutes / 60 seconds = 0.0694 requests per second.

This is all to say that we can safely allocate 300MB of RAM for the system, and 300MB for each customer, comfortably making 2000 requests a day.



$10 /mo

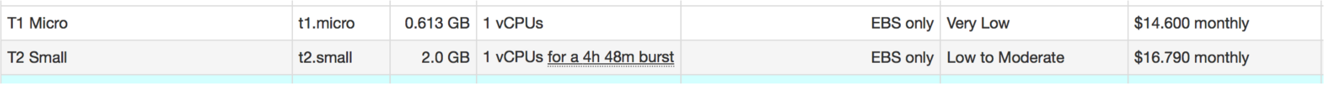
1GB Memory
1 Core Processor
30GB SSD Disk
2TB Transfer

$20 /mo

2GB Memory
2 Core Processor
40GB SSD Disk
3TB Transfer

Digital Ocean droplets may be a good fit at this stage.

This makes a $10 or $20 droplet on DigitalOcean a decent fit for this stage of the project. I could also look at some of AWS’s T instances if I want to sacrifice SSD, and questionable network speed in exchange for use CloudFormation and ECS:



T1 Micro	t1.micro	0.613 GB	1 vCPUs	EBS only	Very Low	$14.600 monthly
T2 Small	t2.small	2.0 GB	1 vCPUs for a 4h 48m burst	EBS only	Low to Moderate	$16.790 monthly

Source: http://www.ec2instances.info/?cost_duration=monthly&selected=t2.small,m1.small

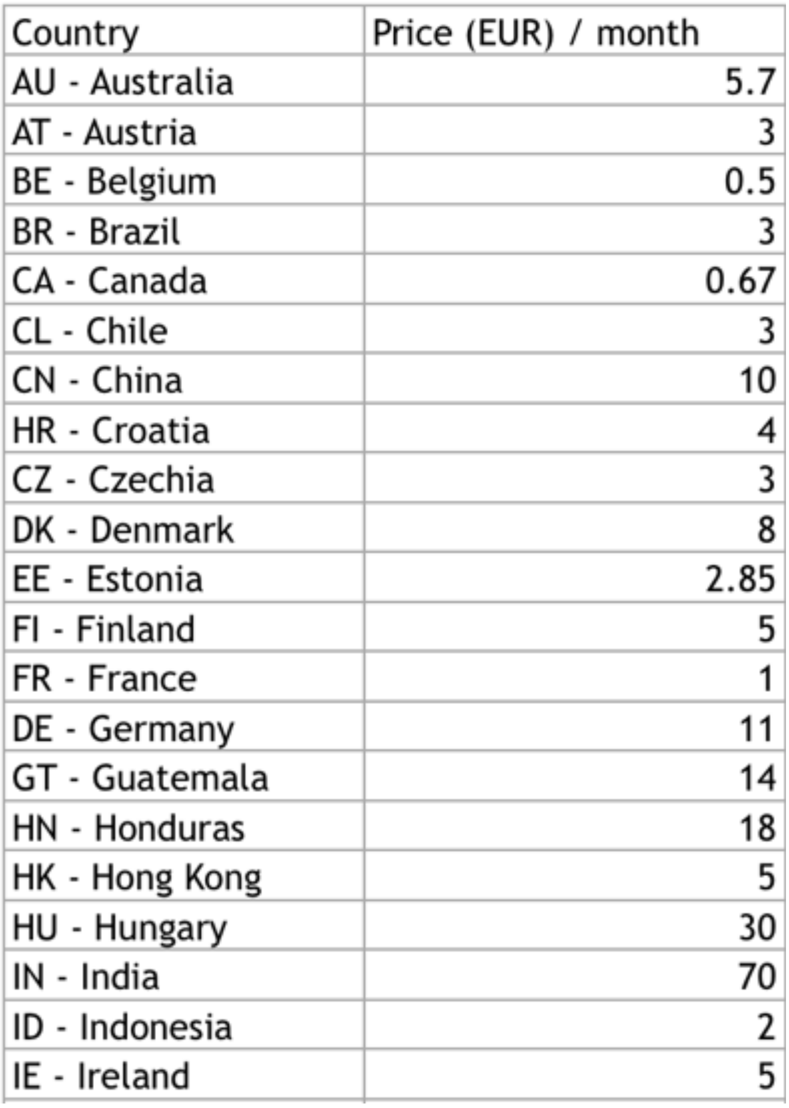
- Since we are hosting this using docker containers, we can use docker containers for redis and our database, just to make things easier for now, so our business costs are essentially just the cost of a server. Let’s put it down as **$15/month**.

Per Customer Costs

We've already calculated that we should be allocating about ~300MB of ram for each new customer, from the EC2 prices, let's say that this will cost about **$4/month**. Our other per customer expenses are:

> SMS Virtual number—to receive messages
>
> SMS Messages—to reply to users
>
> Onboarding—more of a time allocation to get new customers up and running.

There are plenty of services for receiving and replying to SMS messages. For an example, lets look at a pretty popular provider, Nexmo.



Country	Price (EUR) / month
AU - Australia	5.7
AT - Austria	3
BE - Belgium	0.5
BR - Brazil	3
CA - Canada	0.67
CL - Chile	3
CN - China	10
HR - Croatia	4
CZ - Czechia	3
DK - Denmark	8
EE - Estonia	2.85
FI - Finland	5
FR - France	1
DE - Germany	11
GT - Guatemala	14
HN - Honduras	18
HK - Hong Kong	5
HU - Hungary	30
IN - India	70
ID - Indonesia	2
IE - Ireland	5

Some of Nexmo’s virtual numbr costs. They vary greatly!

Cost of virtual number: This will vary greatly from one country to another, making pricing really difficult, as we hope to provide an end to end service, where we fold this cost into the monthly cost of doing business. We may be able to sort out different providers for each country which we hope to support, but for now, let’s take India’s 70 EUR as the going rate, which as of today works out to be ~**$74** USD.

Cost per SMS: Luckily this looks cheaper in India than other services, at 0.008EUR per message, ~$0.01 USD. If we predict 2000 requests per week, that makes 8,000 requests (and therefore SMS messages) a month: **$80**

I’ll keep looking around at different mobile number providers, as we have an important decision to make: Do we want to pass this per-sms charge onto our customers? Or should we absorb it? If we pass it on, there is less risk for us, but we miss an opportunity: if our customer pays for 8000 messages a month and they only use 4000, we make $40! This must be how it feels to be a telco.

The best solution is probably price banding. We can charge in brackets for <8000 messages a month, >8000 messages a month etc. The tricky thing is going to be the cost of virtual number.

Onboarding Costs: Let’s assume that setting all of these things up (at least until we’ve written scripts to do it for us) will take about 8 hours, including some back and forth between myself and clients, virtual number providers etc. If I charge $50/hr for myself, that comes to **$400**, which only has to be paid once per customer (and will cost less once we have grown). Assuming that customers still around for 12 months, that comes to **$33/month** in setup cost.

So our onboarding expenses come to:

```
    4   additional server capacity
 + 74   virtual sms number
 + 80   8000 messages
 + 33   account setup etc.
 = 191
```

$191 a month for an SMS data collection bot?!? That just seems a little expensive for me… That doesn’t even include any margin for us to make any profit. If we wanted just a 20% margin to cover our business costs, plus actually make some money, that would come to: **$229.2**/month. Way too high.

Obviously, we need to find a cheaper SMS provider, otherwise this simply will not work.

https://telerivet.com/pricing—provides an Android app for receiving SMS using an Android phone. Perhaps it would be feasible to receive messages using this application. We could even send them through a different, cheaper service. The only issue is that we have to find somewhere in India (for example with MyWell) that the phone can sit 24/7, with internet access, and fully charged at all times.

Maybe we can provide instructions for our users to set this up… although that seems quite cumbersome, and won’t allow us to scale nicely.

Textlocal also offers more affordable prices, but using shortcodes. This means that our queries would need to start with a keyword:

```
mywell reading of 12.34m yesterday
```

Which may not be too bad, but does take away the ability to ‘chat’ with the system.