

A fully functional Angular dashboard for Beaconstac and a Django Chatbot

BARNIK RAY

MCA 3rd YEAR

EXAM ROLL - **MCA196007**

REG. NO. - **137316** of **2016-17**

MENTOR - Prof. DEBOTOSH BHATTACHARJEE

JADAVPUR UNIVERSITY, KOLKATA

**FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY**

CERTIFICATE OF RECOMMENDATION

This is to certify that the thesis entitled “**A fully functional Angular dashboard for Beaconstac and a Django Chatbot**” has been satisfactorily completed by Barnik Ray(Exam Roll No. - **MCA196007**, Reg. No. - **137316** of **2016-17**). It is a bonafide piece of work carried out under my guidance and supervision and can be accepted in partial fulfilment of the requirements for the Degree of Master of Computer Application, Department of Computer Science and Engineering, Faculty of Engineering and Technology, Jadavpur University, Kolkata.

Prof. Debotosh Bhattacharjee
Mentor
Dept. of CSE, JU

Countersigned

Prof. Mahantapas Kundu
Head of Department
Dept. of CSE, JU

Prof. Chiranjib Bhattacharjee
Dean
Faculty of Engineering & Tech
JU

FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

CERTIFICATE OF APPROVAL

This is to certify that the thesis entitled “**A fully functional Angular dashboard for Beaconstac and a Django Chatbot**” is a bonafide record of work carried out by Barnik Ray in partial fulfilment of the requirements for the award of the degree of Master of Computer Application in the Department of Computer Science and Engineering, Jadavpur University during the period January 2019 to May 2019. It is understood by this approval that the undersigned do not necessarily endorse or approve any statements made, opinions expressed or conclusions drawn there in but approve the thesis only for the purpose for which it has been submitted.

Signature of Examiner

Date :

Signature of Supervisor

Date :

**FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY**

Declaration of Originality and Compliance of Academic Ethics

I hereby declare that this thesis entitled “**A fully functional Angular dashboard for Beaconstac and a Django Chatbot**” contains original research work by the undersigned candidate, as a part of his Degree of Master of Computer Application.

All informations in this document have been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

BARNIK RAY

MCA 3rd YEAR

EXAM ROLL - **MCA196007**

REG. NO. - **137316** of **2016-17**

Signature

Date :

Acknowledgements

I'm extremely grateful to my mentor Prof. Debotosh Bhattacharjee for always extending his support and providing necessary and valuable feedback towards the completion of this project.

I'm also grateful to the developers at Mobstac where I spent a wonderful 5 months as an intern and learnt a lot of things.

Finally I'm grateful to my parents for always extending their support and being on my side throughout. Without their support, it would not have been possible for me to accomplish this task.

Index

<u>Topic</u>	<u>Page No</u>
1. Problem Definition	1
2. Requirements Analysis	
A. Angular Dashboard	2 - 11
B. Django Chatbot	12 - 14
3. Solution/Methodology	
A. Angular Dashboard	15 - 45
B. Django Chatbot	46 - 58
4. Validation Results	59
5. References	60

Problem Definition:

1. Build a functional dashboard for Beaconstac customers based on Angular version 6.

The dashboard features various functionalities, including real-time analytics data received from beacons, NFC tags, and QR codes deployed at multiple physical stores and provide various customisation options for all the products above.

2. A Django chatbot for answering FAQs.

The chatbot is primarily built for answering common FAQs. It is built on Django with VueJS at the frontend.

Requirements Analysis:

The requirements for the applications mentioned above can be categorised into two parts:

1. Angular Dashboard

This problem primarily requires knowledge about Angular framework along with the working principle of beacons, NFC tags, and QR codes.

Described below are a few definitions about proximity marketing, which includes beacons, NFC tags, and QR codes as the primary products:

- What is proximity marketing?

Proximity marketing refers to communicating with customers at the right place, the right time and with highly relevant and personalized messages, on their smartphones - be it greeting at the entry points, special offers in the store aisles, or getting feedback on a new product.

Beacons

- What is a Bluetooth beacon?

A beacon is a small Bluetooth radio transmitter, powered by batteries. Beacons are similar to a lighthouse in functionality. These small hardware devices incessantly transmit Bluetooth Low Energy (BLE) signals.

- What are the types of beacons?

- *Battery powered*

Bluetooth beacons operate using the Bluetooth 4.0 Low Energy standard, so battery powered devices are possible. The battery life of devices varies depending on the manufacturer. The Bluetooth LE protocol is significantly more power efficient than Bluetooth Classic.

Several chipsets makers, including Texas Instruments and Nordic Semiconductor, now supply chipsets optimized for iBeacon use. Power consumption depends on iBeacon configuration parameters of advertising interval and transmits power. Battery life can range between 1–48 months. Apple's recommended setting of 100 ms advertising interval with a coin cell battery provides for 1–3 months of life, which increases to 2–3 years as advertising interval is increased to 900 ms

- *USB powered*

Bluetooth beacons can also come in the form of USB dongles. These small USB beacons can be powered by a standard USB port which makes them ideal for long term permanent installations.

- The two most commonly used beacon protocols:
- **iBeacon:** iBeacon is a protocol developed by Apple and introduced at the Apple Worldwide Developers Conference in 2013. Various vendors have since made iBeacon-compatible hardware transmitters – typically called beacons – a class of Bluetooth low energy(BLE) devices that broadcast their identifier to nearby portable electronic devices. The technology enables smartphones, tablets, and other devices to perform actions when near an iBeacon.
- iBeacon is based on Bluetooth low energy proximity sensing by transmitting a universally unique identifier picked up by a compatible app or operating system. The identifier and several bytes sent with it can be used to determine the device's physical location, track customers, or trigger a location-based action on the device such as a check-in on social media or push notification.
- iBeacon can also be used with an application as an indoor positioning system, which helps smartphones determine their approximate location or context. With the help of an iBeacon, a smartphone's software can approximately find its relative location to an iBeacon in a store. Brick and

mortar retail stores use the beacons for mobile commerce, offering customers exclusive deals through mobile marketing, and can enable mobile payments through point of sale systems.

- **Eddystone:** Eddystone is a Bluetooth Low Energy beacon profile released by Google in July 2015. The Apache 2.0-licensed, cross-platform, and versioned profile contains several frame types, including Eddystone-UID, Eddystone-URL, and Eddystone-TLM. Eddystone-URL is used by the Physical Web project, whereas native apps typically use Eddystone-UID on a user's device, including Google's first-party apps such as Google Maps.
- How are beacons used in proximity marketing?

The Bluetooth enabled smartphones are capable of scanning and displaying these signals. Beacons could be deployed on store-fronts, real estate properties, amusement parks, events, and other public venues to broadcast contextually-relevant advertisements and notifications.

The primary goal of the beacons section of the Beaconstac dashboard is to provide a platform for the users to customize the beacons to display various campaigns and also to view analytics information regarding the number of scans, number of campaign visits, etc.

NFC

- What is NFC?

NFC stands for Near Field Communication, and this technology allows two devices, or a device and a physical object to communicate without having to set up a prior connection. This device can be a smartphone, tablet PC, digital signage, smart posters, and intelligent signs.

NFC technology has been in the payment section for quite some years now, but in recent years, forward-looking marketers and advertising professionals are leveraging NFC for marketing and delivering more productive and more interactive customer experiences.

- What are NFC tags, and how do they work?

NFC tags are passive devices, which means that they operate without a power supply of their own and are reliant on an active device to come into range before they are activated. The trade-off here is that these devices can't do any processing of their own. Instead, they are simply used to transfer information to an active device, such as a smartphone.

To power these NFC tags, electromagnetic induction is used to create a current in the passive device. We won't get too technical on this, but the basic principle is that coils of wire can be used to produce electromagnetic waves, which can then be picked up and turned back into current by another coil of

wire. This is very similar to the techniques used for wireless charging technologies, albeit much less powerful.

The active devices, such as your smartphone, are responsible for generating the magnetic field. This is done with a simple coil of wire, which produces magnetic fields perpendicular to the flow of the alternating current in the cable. The strength of the magnetic field can be adjusted by varying the number of turns in the wire coil, or increasing the current flowing through the wire.

However, more current requires more energy, and very high power requirements would not be desirable for use in battery-powered mobile technologies. Hence why NFC operates over just a few inches, rather than the many meters that we're used to with other types of wireless communication.

The passive device works in the same way, just in reverse. Once the passive device is in range of the active device's magnetic field, the electrons in the receiving coil of wire begin to produce a current that matches that in the transmitting smartphone. There is always some power lost during transmission through the air, but over short distances, the current generated is enough to power the circuitry in the NFC tag.

These circuits are fine-tuned to a certain frequency, which increases the device's sensitivity to charging frequencies. This allows for a maximum transfer of energy across the air. Once the tag is powered up, it can sync up and send data over the 13.56MHz NFC transmission frequency at either 106, 212 or 424 Kbps, just like your regular NFC communication between phones or other larger devices.

- What are the different types of NFC tags?

NFC tags communicate using the ISO 14443 type A and B wireless standards, which is the international standard for contactless smart cards, used on many public transportation systems. This is why NFC devices can be used with existing contact-less technologies, such as card payment points.

There is a range of different tag types available, each offering different storage levels and transfer speeds. Tag types 1 and 2 come with capacities between just a tiny 48 bytes and 2 kilobytes of data and can transmit that information at only 106 kbit/s.

Although that may sound quite small, especially compared to your typical SD card, that's enough data for some simple pieces of information, such as a website URL, and is all you need for most basic NFC tags. These tags are designed to be highly cost-effective, and can also be re-used if you want to change the data stored on them.

- What is NFC marketing?

NFC marketing is a proximity marketing channel used to interact with a location or a physical object at a small distance. NFC tags/stickers are embedded on OOH displays or products. NFC technology enables a seamless exchange of information with just a tap. Let's discuss what NFC technology is, how does it work & how can marketers leverage it.

QR Codes

- What are QR codes?

A QR code is a 2-D barcode that can be digitally read by smartphones. QR codes contain information just like barcodes. They can be used to make a call, send a message or email, or even open a website.

Here is an entire list of things we can do with a QR code:

1. Add contact details: QR codes serve as a vCard and can help you add and share contact information in seconds.
2. Send predefined texts: Did you know you can scan a QR code and send texts with the recipient and contents of the message already added? Once you scan a text QR code, all you have to do is press send.
3. Send preset emails: Just like texts, you can also send predefined emails. Again, scanning the QR code will open up your email, and you can hit send.
4. Make a call: QR codes can also be used to call someone. Businesses can use this to have you call them or customer support with just one scan.
5. Reveal discounts and coupon codes: What if you didn't have to remember long promo codes? With a single scan of a QR code, you can be redirected to the online checkout with the promo code pre-applied.
6. Navigate to the store: Some businesses even use QR codes on Out-of-Home advertising to help you navigate to your nearest store.

7. Add calendar events: An urgent sale coming up? Remind yourself with a calendar event triggered by the scan of a QR code.
8. Connect to WiFi without password: You no longer need to share or remember passwords. Scan the QR code and join the WiFi network.
9. Engage with social media: Scan QR codes and instantly follow, like, and leave a review on social media business pages.
10. Download apps: You can now scan QR codes to download an app in your respective App Store.

- The two basic types of QR codes:

- **Static QR codes:**

A static QR code is a QR code that once generated, cannot be overwritten. It can store information which is not editable. Static QR codes find a lot of use in instances where either the information does not need to be updated, or it is for a one-off campaign. You can create different types of static QR codes -

- Video QR code
- Event QR code
- Image Gallery QR code
- Poster QR code
- Flyer QR code
- Business card QR code
- vCard QR code
- Website QR code

- Facebook QR code
- Location QR code
- Billboard QR code

- **Dynamic QR codes:**

Dynamic QR codes are QR codes which are editable in real time with Beaconstac's intuitive QR code solution. Not only can you update small details, but you can also change the entire URL and avoid pesky 404 errors.

Dynamic QR codes mean that you no longer need to reprint the QR code. Beaconstac's solution also allows you to convert potential leads who have scanned the QR code but have not taken any action yet with online retargeting on Google and Facebook.

- What is QR code marketing?

QR Code marketing is a proximity marketing channel that leverages quick response (QR) codes to bridge the gap between offline and online channels. These QR codes are put up in strategic locations for consumers to scan using their smartphone and access information about the product, brand, or offers.

2. Django Chatbot

The Django chatbot is built on Django with VueJS as the template driver. In this project, I have tried to create a useful chatbot. The idea was to build a chatbot, which can answer particular questions from various domains as well as answer custom questions asked by the user.

I have successfully built a chatbot which can perform the operations above and satisfy all the conditions along with a beautiful visual appearance.

In a nutshell, the chatbot is efficient, elegant, and user-friendly.

Approach to the problem:

Efficient:

I have approached this problem from the perspective of the end users and tried to build an interactive and responsive experience from the very beginning of building this bot. At the same time, we have also focused on the efficiency and accuracy of the bot, which is one of the most important functional features of the bot.

Fast:

Besides being responsive and accurate, a bot should be fast in responding to the queries of the user. Latency is a significant factor when it comes to the experience of an interactive app.

User-friendly:

We have tried to make our bot extremely user-friendly besides being informative. For this, we did a lot of research to choose the right technology stack, which will serve all the conditions mentioned above.

Technology Stack used for building the bot:

Backend Driver: I decided to go for Django as the primary backend framework for the bot. Django is one of the most commonly used backend frameworks in this era, and it is particularly preferred for its robust yet straightforward architecture as well as fewer LOCs required to perform critical and complex tasks. I have used Django 2.2.1 for building this bot.

Frontend Driver: As the frontend driver, I chose VueJS as the template engine as it is an extremely feature rich yet lightweight framework and hence has very minimal hardware requirements both in terms of development and deployment.

RASA Stack: For processing the questions asked by the users to the bot they should be passed through a NLP(Natural Language Processing) unit so that the necessary features of the question can be extracted. This is important because the accuracy of the answers depend upon how well the features can be extracted from the question so that the necessary intents can be identified and can be sent for the retrieval of the answer. For this, I chose the RASA stack, which comprises of the RASA NLU and the RASA core, which performs

the task above. It is an excellent open source framework which has great tools right from training the NLU unit to the execution of the model.

Google Custom Search API: After extracting the features of the question, the answer needs to be retrieved for the same. For this, I chose the google custom search engine which is a custom engine fully customizable to suit one's needs and provides speedy responses which improves the overall user experience while using the bot.

Heroku: From the very beginning of building this bot I wanted to make it production ready so that it can be tested in real time by the users. After doing some research, I decided to go with the heroku cloud platform to deploy my bot. Heroku is an excellent platform with a blazing fast server TTR(Time to Response) which is necessary as there will be many API calls throughout the lifecycle of the bot.

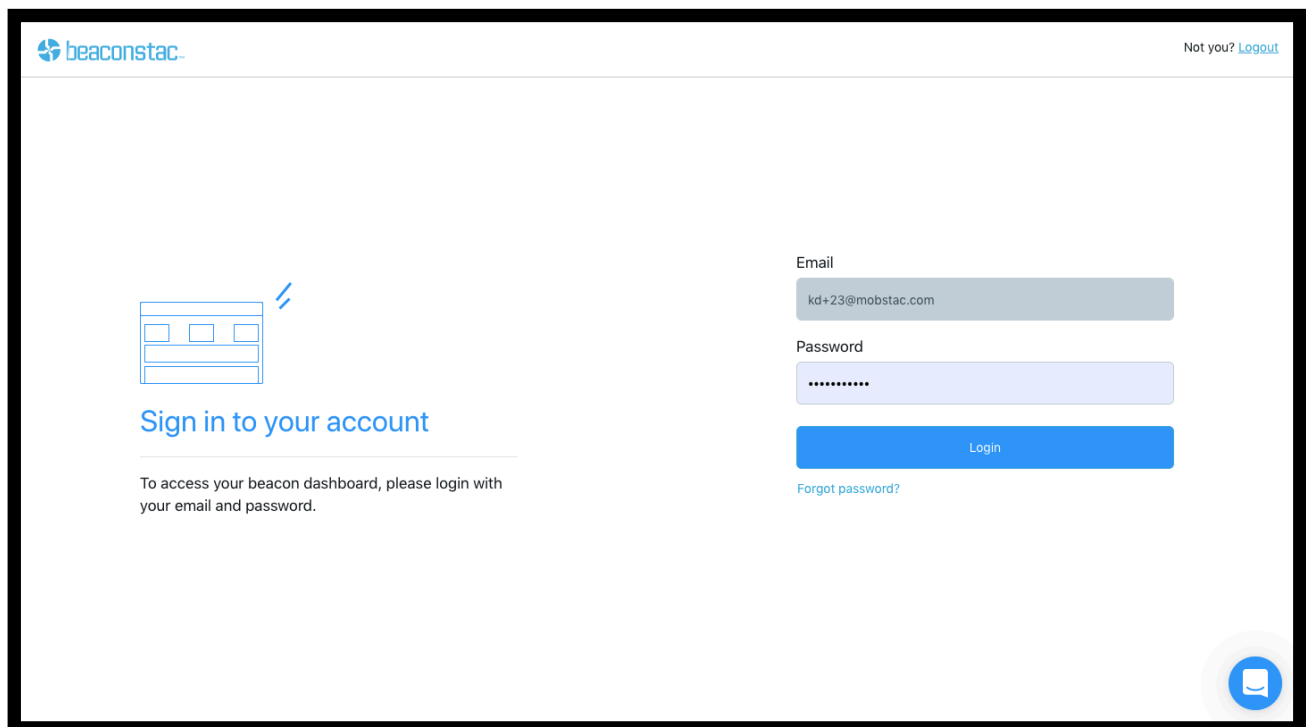
Solution/Methodology:

The solution phase is divided further into two parts. Let us first start with the Angular dashboard:

Angular Dashboard:

Let us first explore the design of the dashboard, which is one of the most important features while designing any frontend application.

A dashboard should be designed in such a way that it is intuitive, responsive, and has a nice UX/UI, which will provide the users with a great experience.

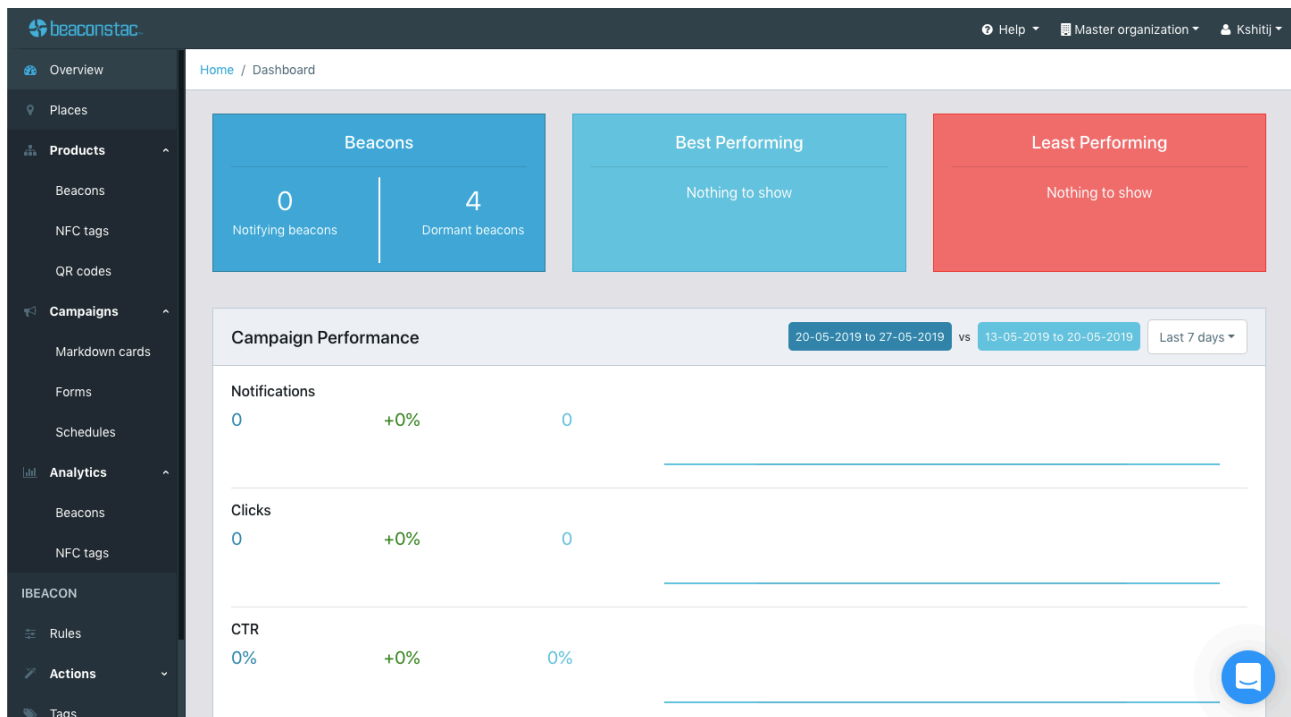


The login page of the Beaconstac dashboard

The basic design of the Beaconstac dashboard is also based on the same principles.

The dashboard has the following headers on the left side navigation bar:

1. *Overview*: This section is the default section which opens up when the user logs in to the dashboard. It provides an overall summary of the beacons/NFC tags/QR codes depending upon which product the user is subscribed to. The summary includes the following data:

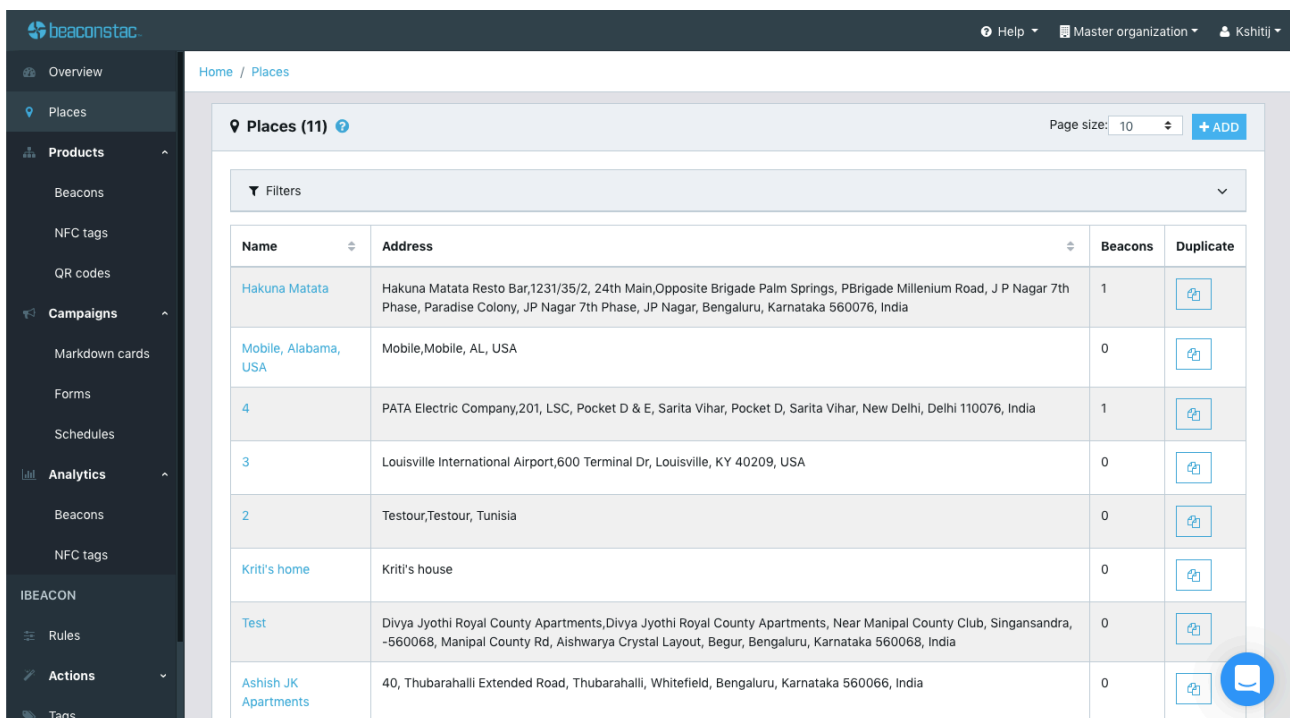


The overview screen of the beaconstac dashboard after login

- For beacons it shows the active and inactive beacons, the best and the least performing beacons.

- It also displays an analytics graph which provides the usage statistics for notifications sent, notifications clicked, and CTRs redirected over a period which is customisable.
- For NFC tags it displays the same info as beacons
- For QR codes it shows the active and inactive dynamic QR codes, the number of QR code scans over some time and related analytics data.

2. *Places*: The places tab contains information about the geographical locations where the beacons/NFC tags/QR codes are deployed. Before deploying any of these products, the places need to be defined, and each of the products must be attached to any one of those places.



Name	Address	Beacons	Duplicate
Hakuna Matata	Hakuna Matata Resto Bar,1231/35/2, 24th Main,Opposite Brigade Palm Springs, PBrigade Millenium Road, J P Nagar 7th Phase, Paradise Colony, JP Nagar 7th Phase, JP Nagar, Bengaluru, Karnataka 560076, India	1	
Mobile, Alabama, USA	Mobile,Mobile, AL, USA	0	
4	PATA Electric Company,201, LSC, Pocket D & E, Sarita Vihar, Pocket D, Sarita Vihar, New Delhi, Delhi 110076, India	1	
3	Louisville International Airport,600 Terminal Dr, Louisville, KY 40209, USA	0	
2	Testour,Testour, Tunisia	0	
Kriti's home	Kriti's house	0	
Test	Divya Jyothi Royal County Apartments,Divya Jyothi Royal County Apartments, Near Manipal County Club, Singansandra, -560068, Manipal County Rd, Aishwarya Crystal Layout, Begur, Bengaluru, Karnataka 560068, India	0	
Ashish JK Apartments	40, Thubarahalli Extended Road, Thubarahalli, Whitefield, Bengaluru, Karnataka 560066, India	0	

The places screen of Beaconstac dashboard

3. *Products*: This is the main section of the dashboard, which contains information about the different products.

- The products section has separate sections for beacons, NFC tags and QR codes which show depending upon which products the user is subscribed to.
- Each section is further subdivided into subsections for each of the mentioned products. The subsections contain customised controls and options for each of the individual products.

Beacons

Home / Beacons

Beacons (6) ? Page size: 10

Filters

<input type="checkbox"/>	Name	Organization	Place	Serial Number	Campaign	Last Seen	Billing State
<input type="checkbox"/>	Super 123	Local Business 2	RTO 1, 2, 3	0117C597A9E9	Markdown card	May 27 2019 17:29	Enabled
<input type="checkbox"/>	Shiny New - A9E9 - da	Local Business 2	RTO 1, 2, 3	AC233F235365	Markdown card	Jun 13 2017 18:00	Enabled
<input type="checkbox"/>	Library	Master organization	RTO 1, 2, 3	0117C55F568E	Form	Not yet seen	Enabled
<input type="checkbox"/>	Engg	Master organization	Hakuna Matata	0117C557D063	Markdown card	Oct 22 2018 16:00	Enabled
<input type="checkbox"/>	Conf Room	Master organization	4	0117C55E7D5F	Custom URL	Jun 05 2018 12:08	Enabled
<input type="checkbox"/>	12, Taylor Street, Ilford - Sale Sign	Master organization	RTO 1, 2, 3	0117C55A185D	Markdown card	Not yet seen	Enabled

Prev 1 Next

© 2019 MobStac, Inc. Phone support: +1 (646) 965-3378

The beacons screen of the dashboard

- For beacons, it contains the details of the beacons like its serial number, type. It also contains the places where they are deployed along with their last scan date and time and the respective campaigns they are running.
- On clicking a beacon, the following customisations are available:

The first subsection is the beacon details section:

The screenshot displays the 'Beacon Details' section of the beaconstac dashboard. The interface includes a sidebar with navigation options and a main content area. The 'Beacon Details' form contains the following fields and sections:

- Name***: Super 123
- Serial Number**: 0117C597A9E9
- Place**: RTO 1, 2, 3
- Associated Tags**: iBeacon Campaign, Eddystone Campaign
- Eddystone Notifications**: Eddystone notifications help users to discover what's around them, by surfacing location-specific notifications for apps and websites. Customize the appearance of your notification here.
- Language**: English (default)
 - ☒ Set default language
- Title***: Local Business 2 Campaign

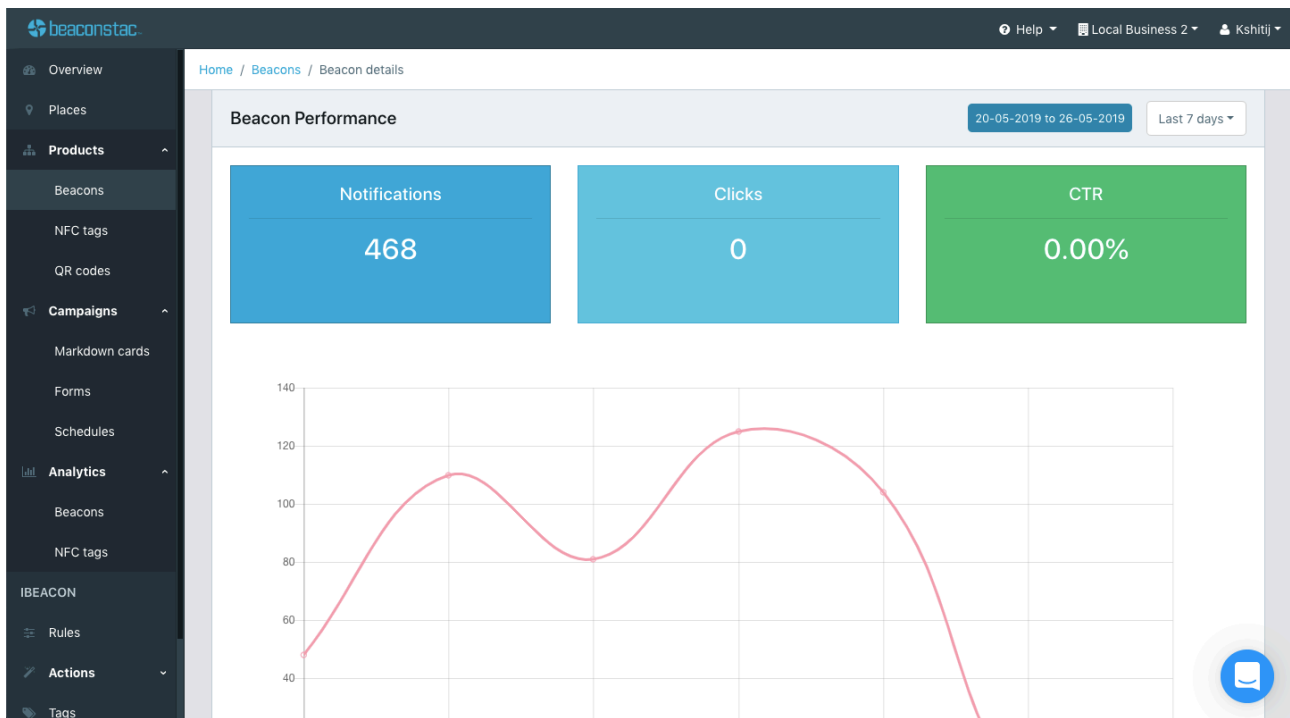
A 'Disable' button is located in the top right corner of the form area.

The beacon details section of the beaconstac dashboard

- The name and place of the beacon can be changed
- The tags associated with the beacons can be changed, and new tags can be added.
- There are separate sections for iBeacon and Eddystone campaigns.

- The iBeacon campaigns show the rules and the associated actions with the rules.
- The Eddystone campaigns show the type of campaigns running on the beacons. The following campaigns are available for the beacons:
 - Custom URL – redirects to a custom URL
 - Markdown Card – A custom poster which can be created in the markdown cards section discussed below.
 - Form – A custom made a form which can be created to serve purposes like taking feedback from the users.
 - Schedule – Run a schedule on the beacons, which can be configured in the schedules section discussed below.
- Eddystone notifications: Eddystone notifications help users to discover what's around them, by surfacing location-specific notifications for apps and websites.
 - The following customisation options are available for Eddystone notifications.
 - Language – The default language of the notification can be set here
 - Title – The title of the notification.
 - Description – The description of the notification.
 - Icon – The icon of the notification.
 - Cover – The cover photo to be displayed in the notification. This photo can be customised for both portrait and landscape viewing.
 - All these features can be edited and changed.

The second subsection is the beacon performance section:

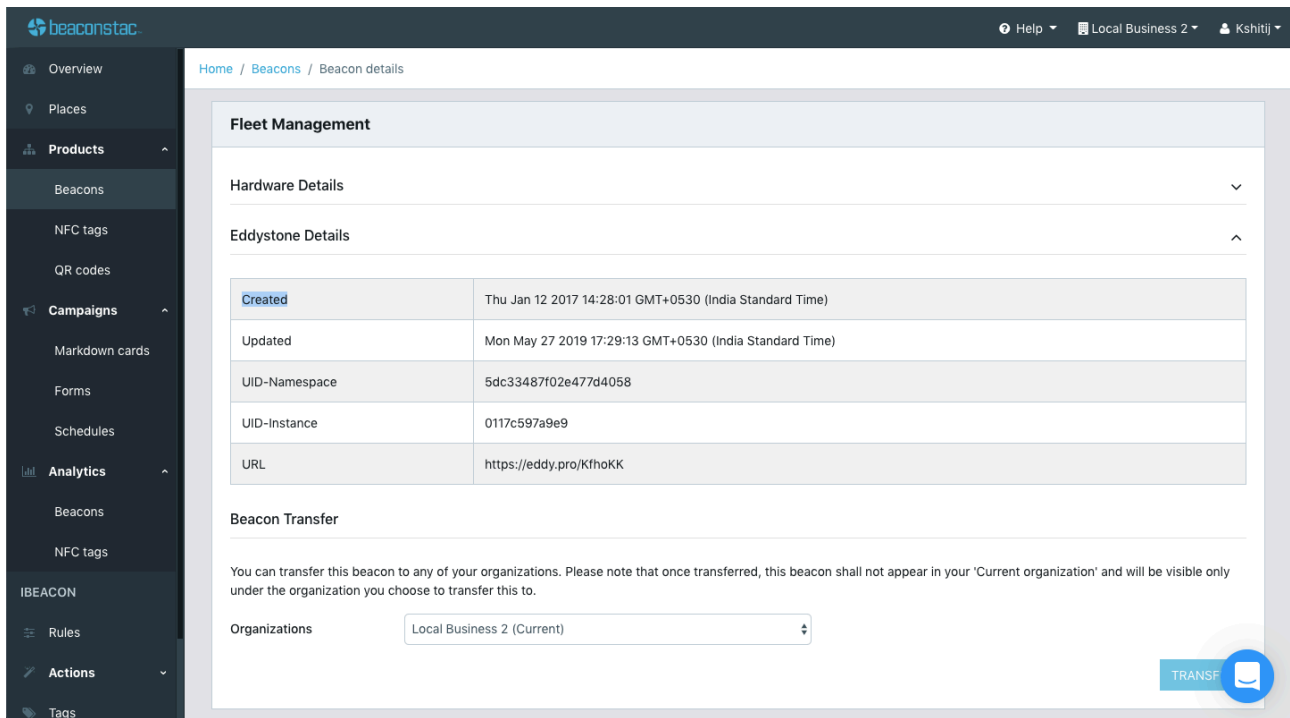


The beacon performance section of the dashboard

This subsection gives information regarding the performance of the beacon.

- It shows the total number of notifications sent by the beacons, the clicks on the notifications and the CTR(Click Through Rate) % to those clicks.
- It also displays a usage graph below over the last seven days which plots all of the above information. The time limit of the usage statistics can be changed.

The third subsection is the fleet management section, which offers the following options:

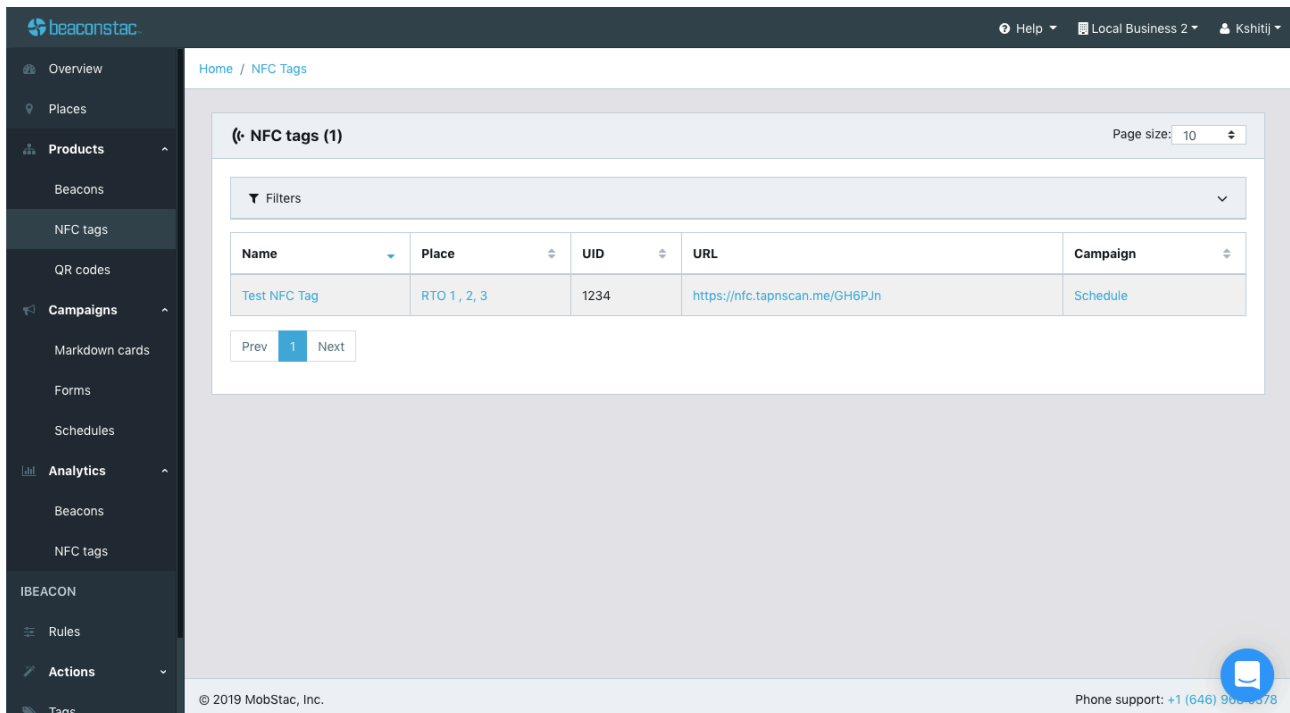


The fleet management section of the dashboard

- Hardware details: This provides the hardware details of the beacons whose configurations can be changed using the Beaconstac app, which is available on both Android and iOS.
- Eddystone details: This section contains the information about the Eddystone configuration of the beacons like the date created, date updated, and the Eddystone URL.
- The third section provides options for beacon transfer, where the beacon ownership can be transferred to any of the user organisations.

The beacon section also contains the option to 'disable' a beacon at the top and even to permanently 'delete' a beacon from the organisation. Once a beacon is deleted, it cannot be restored.

NFC



The screenshot shows the beaconstac dashboard with a dark sidebar on the left containing navigation links: Overview, Places, Products, Beacons, NFC tags, QR codes, Campaigns, Markdown cards, Forms, Schedules, Analytics, Beacons, NFC tags, IBEACON, Rules, Actions, and Tags. The main content area is titled 'Home / NFC Tags' and displays '(NFC tags (1)'. A 'Page size: 10' dropdown is in the top right. Below a 'Filters' dropdown is a table with columns: Name, Place, UID, URL, and Campaign. The table contains one row: 'Test NFC Tag', 'RTO 1 , 2, 3', '1234', 'https://nfc.tapnscan.me/GH6PJn', and 'Schedule'. A pagination bar shows 'Prev', '1', and 'Next'. The footer includes '© 2019 MobStac, Inc.' and 'Phone support: +1 (646) 900-4378' with a chat icon.

Name	Place	UID	URL	Campaign
Test NFC Tag	RTO 1 , 2, 3	1234	https://nfc.tapnscan.me/GH6PJn	Schedule

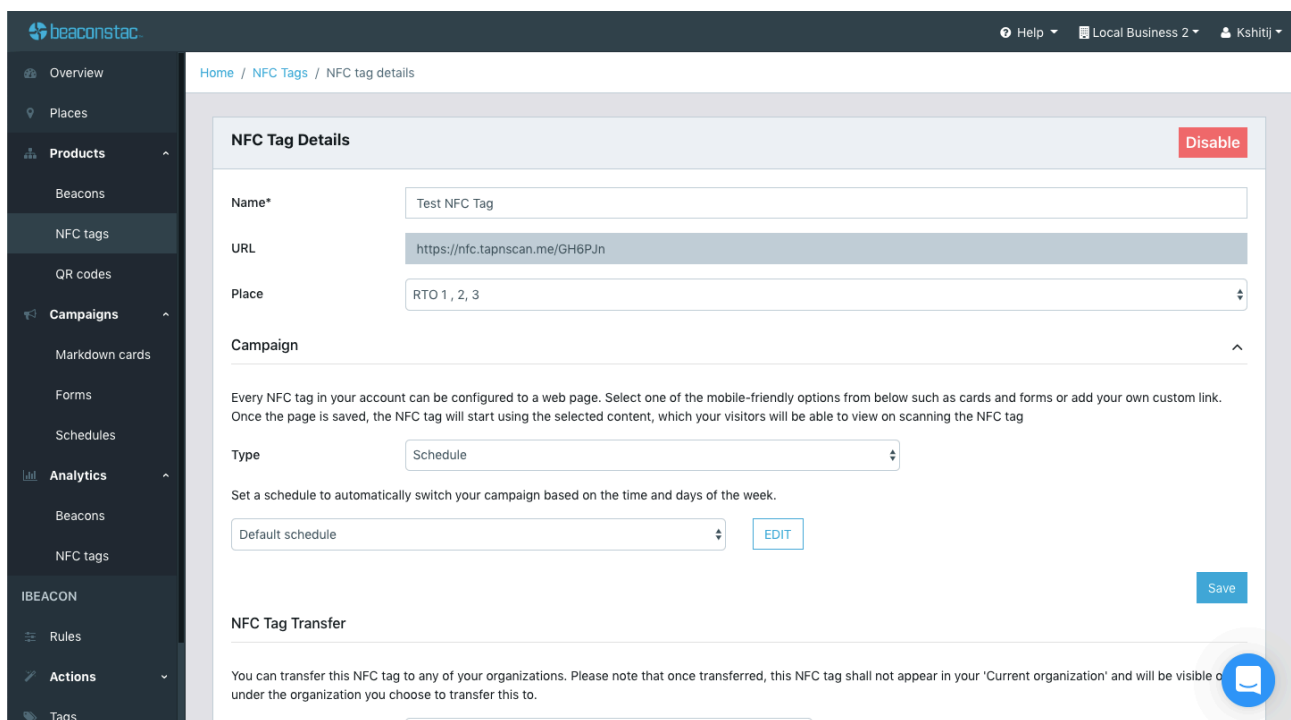
The NFC tags section of the dashboard

Now let us come to the NFC tags section.

- Similar to beacons for NFC tags section contains the info regarding the details of the tag like the UID of the tag, the URL of the tag campaign along with the campaign running on the tag.

On clicking a NFC tag the following options appear:

The first subsection is the NFC tag details section:

The screenshot shows the Beaconstac dashboard interface. On the left is a dark sidebar with navigation links: Overview, Places, Products (with a sub-menu for Beacons, NFC tags, and QR codes), Campaigns (with sub-menu items: Markdown cards, Forms, Schedules), Analytics (with sub-menu items: Beacons, NFC tags), IBEACON, Rules, and Actions. The main content area is titled 'NFC Tag Details' and includes a 'Disable' button in the top right. The form contains fields for 'Name*' (filled with 'Test NFC Tag'), 'URL' (filled with 'https://nfc.tapscan.me/GH6PJn'), and 'Place' (a dropdown menu showing 'RTO 1, 2, 3'). Below these is a 'Campaign' section with a heading and a paragraph explaining that NFC tags can be configured to a web page. It features a 'Type' dropdown menu set to 'Schedule' and a section for setting a schedule to automatically switch campaigns based on time and days of the week, with a 'Default schedule' dropdown and an 'EDIT' button. At the bottom right of the form is a 'Save' button. A final section titled 'NFC Tag Transfer' contains a paragraph explaining that the tag can be transferred to other organizations and will no longer be visible in the current one. A blue chat bubble icon is visible in the bottom right corner of the dashboard.

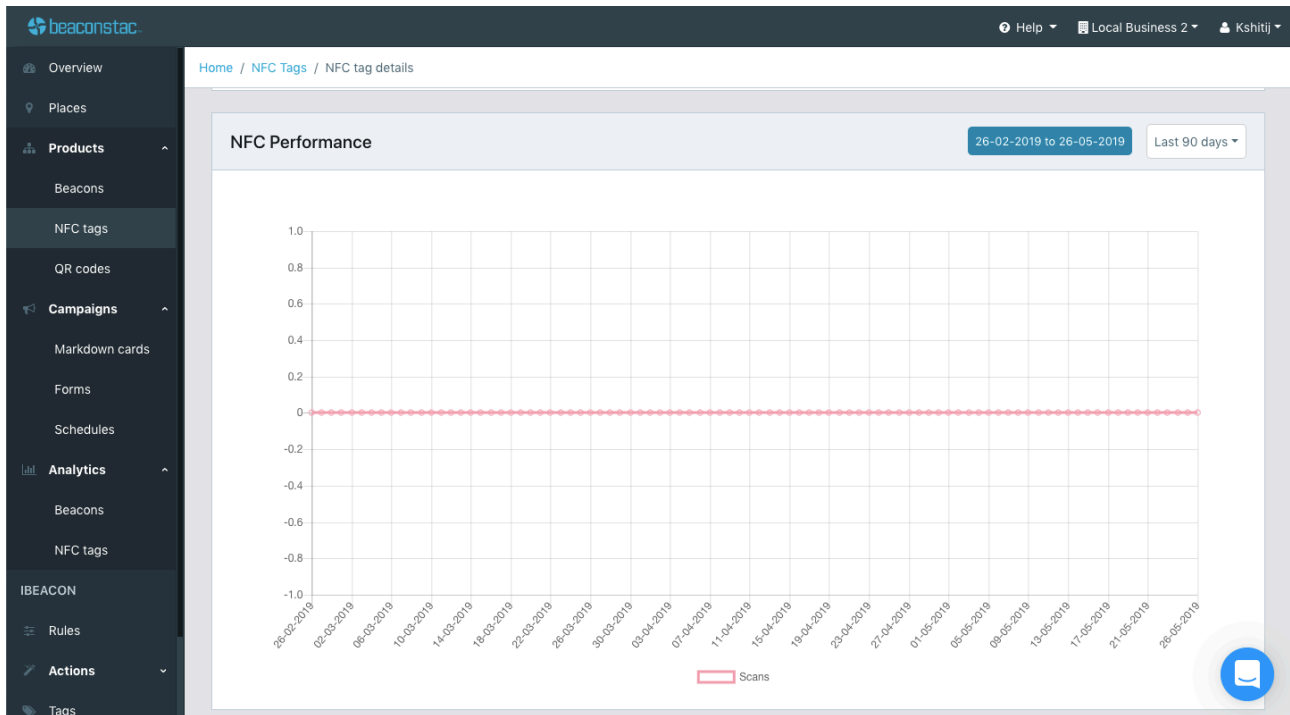
NFC tag details section of the dashboard

- This section provides customisation options like name and place of the NFC tag similar to beacons.
- It also has a campaigns section which allows the user to choose the campaigns running on the NFC tags.
- The following campaigns are available for the NFC tags:

Custom URL, Markdown card, Schedule and Form similar to beacons

- There is also a section for NFC Tag transfer where NFC tags can be transferred to any of the user's organisations.

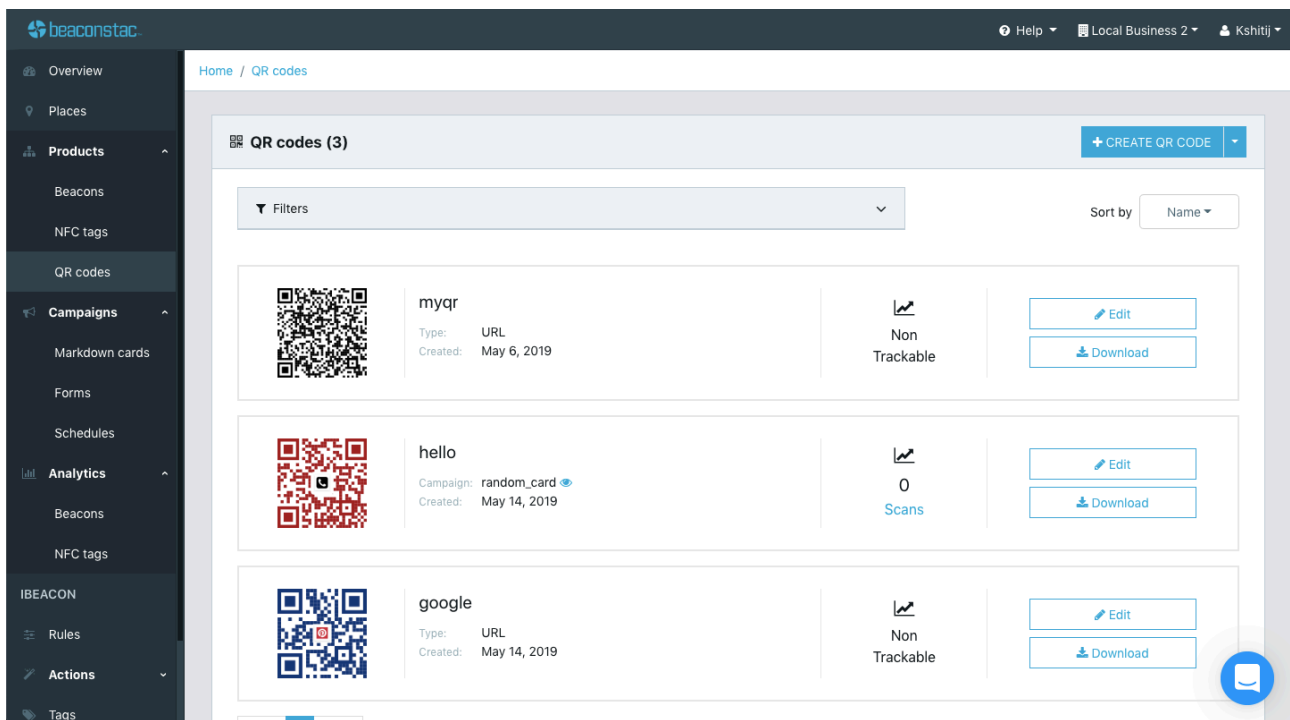
The second subsection for NFC tags is the Analytics section:



The analytics section of the NFC tags section

This section provides scan data in graphical format depicting the performance of the tag. The graph can be viewed for various timelines.

QR Codes



The QR codes section of the dashboard

The next major subsection under products is that of QR codes:

At the top, there is a Create QR code option which has options for creating both static and dynamic QR codes.

- Both static and dynamic QR codes can be created with the following options:
- The name of the QR code.
- The campaigns the QR codes will run if it is a dynamic QR code. The campaign can be set to one of Custom URL, Markdown card, Schedule and Form similar to beacons and NFC tags.
- The color of the QR code can be set.

- The QR code can be customized with a logo at the center.

The screenshot displays the 'Create QR' interface in the beaconstac dashboard. The sidebar on the left lists navigation items: Overview, Places, Products (with sub-items Beacons, NFC tags, and QR codes), Campaigns (with sub-items Markdown cards, Forms, and Schedules), Analytics (with sub-items Beacons and NFC tags), and IBEACON (with sub-items Rules and Actions). The main panel is titled 'Create QR' and includes a 'Name*' input field, a 'Campaign' section with a dropdown menu set to 'No campaign', and a 'Customize' section with a 'Color' selector and a 'Logo (100x100)' area. The logo area features a 'Drag and drop' instruction and a 'Choose from gallery' section with icons for various social media and apps. A 'Preview' section is located at the bottom of the main panel.

The create QR section of the dashboard

- After reviewing the code the user can choose the size of the QR code from one of 128px, 256px, 512px, 1024px and 2048px and the image format of the QR code from one of PDF, PNG, SVG, and JPEG.
- The user can then proceed to download/save the QR code.
- There is also a QR transfer option below to transfer the QR code to any of the user's organisations.
- For QR codes, it contains the types of QR codes along with the usage and scan statistics for the dynamic QR codes as static QR codes cannot be tracked. It also displays the actual QR code beside each code and also provides options to edit and download the QR code.

On clicking the 'edit' QR codes the following options are available:

The first section is the edit QR section:

- Here the name type and website of the QR code is possible, and the name can be changed.
- The QR code can be downloaded in one of the sizes mentioned above and image formats.
- There is also an option to delete the QR code at the top.

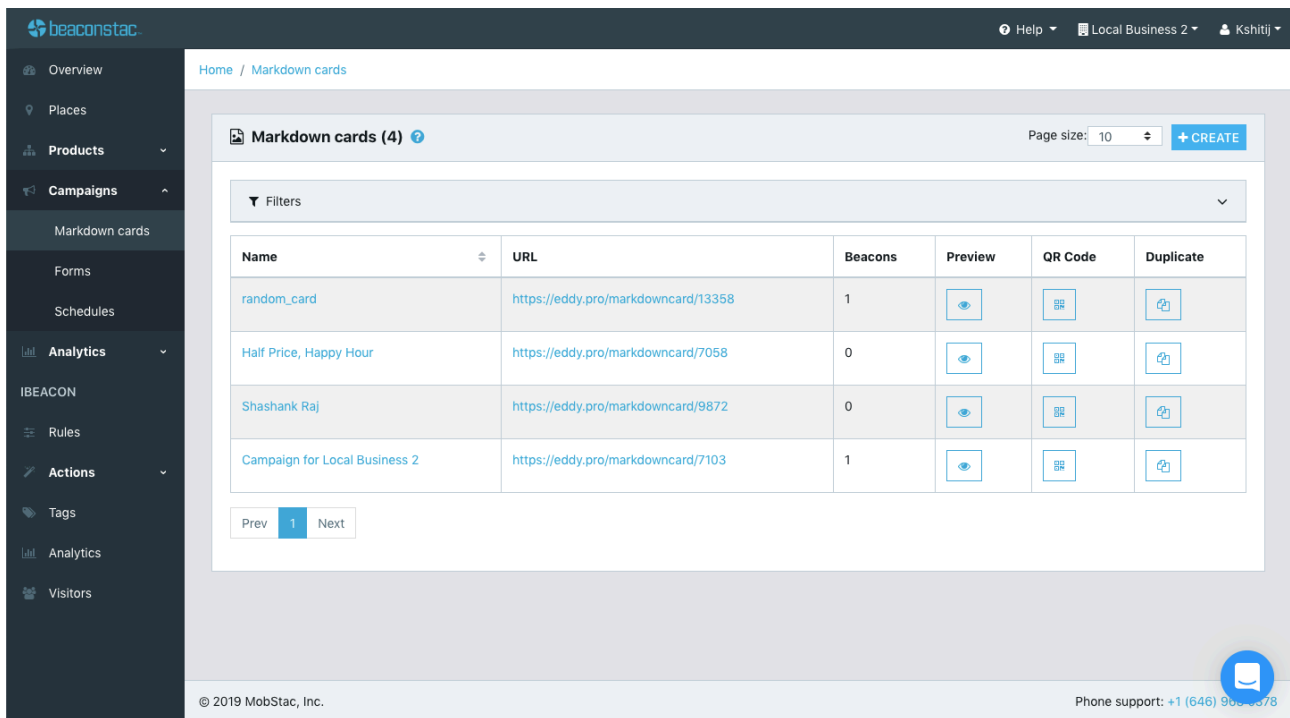
For dynamic QR codes, the following additional options are available:

- The color and the logo of the QR code can be changed.
- The campaign running on the QR code can be changed
- The QR code can be transferred to another organisation.
- There is a QR performance section which gives the scan details for the dynamic QR codes over a period of time which can be modified.

4. *Campaigns*: The next significant section after products is campaigns. This section is dedicated to the type of campaigns the products are running, and they can be changed, customised, and added.

There are primarily three campaigns which need configuration:

1. *Markdown cards*: This section contains custom designed posters which the user can choose from a set of templates or create using simple markdown language used in .md files.



The screenshot displays the Beaconstac dashboard interface. On the left is a dark sidebar with navigation links: Overview, Places, Products, Campaigns (expanded), Analytics, IBEACON, Rules, Actions, Tags, Analytics, and Visitors. The main content area is titled 'Home / Markdown cards'. At the top of this section, it says 'Markdown cards (4)' with a help icon, a 'Page size: 10' dropdown, and a '+ CREATE' button. Below this is a 'Filters' dropdown. A table lists four markdown cards:

Name	URL	Beacons	Preview	QR Code	Duplicate
random_card	https://eddy.pro/markdowncard/13358	1			
Half Price, Happy Hour	https://eddy.pro/markdowncard/7058	0			
Shashank Raj	https://eddy.pro/markdowncard/9872	0			
Campaign for Local Business 2	https://eddy.pro/markdowncard/7103	1			

Below the table is a pagination control showing 'Prev', '1' (selected), and 'Next'. The footer of the dashboard includes '© 2019 MobStac, Inc.' on the left and 'Phone support: +1 (646) 906-8878' on the right, next to a chat icon.

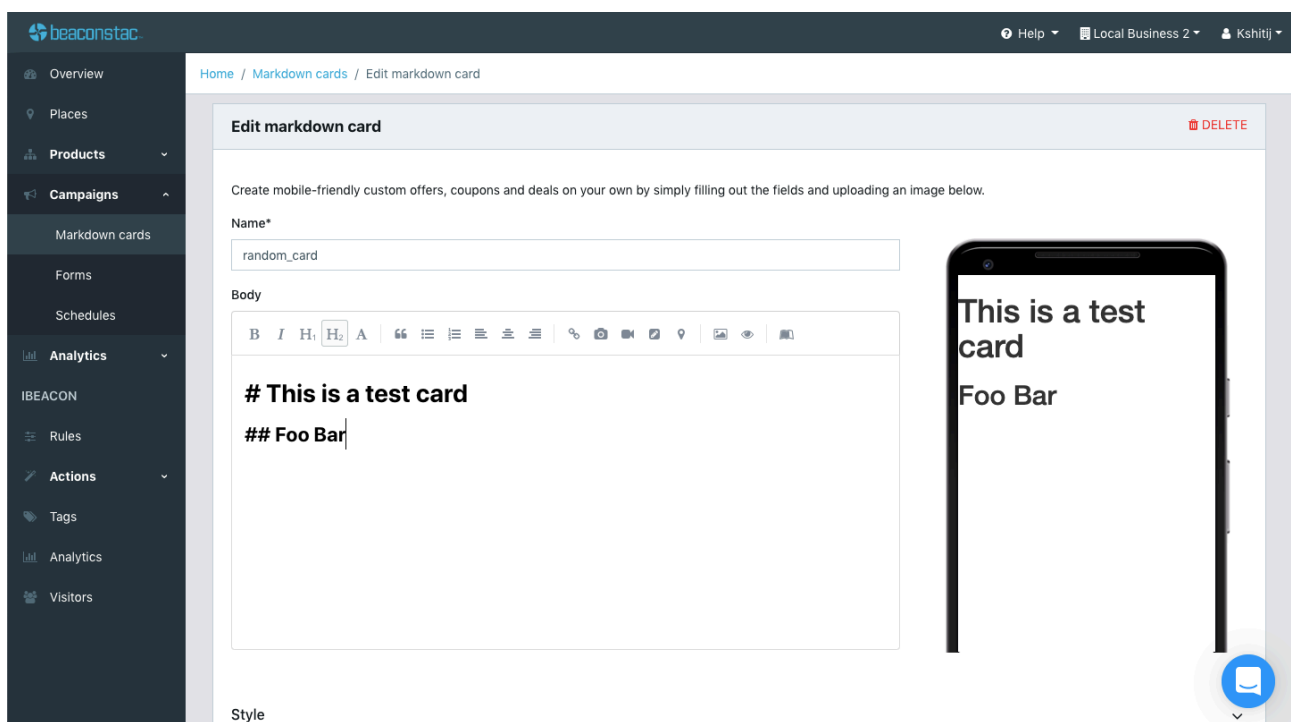
The markdown card section of the dashboard

On clicking this section, it displays the markdown cards which are presently there in the user's account.

- It provides details like name and URL of the markdown card, the products it is attached to. It can also be previewed, and there is also an option to create a QR code for it. Also, there is an additional option to duplicate the markdown card.

On clicking the markdown card, the following sections are displayed:

Edit markdown card:



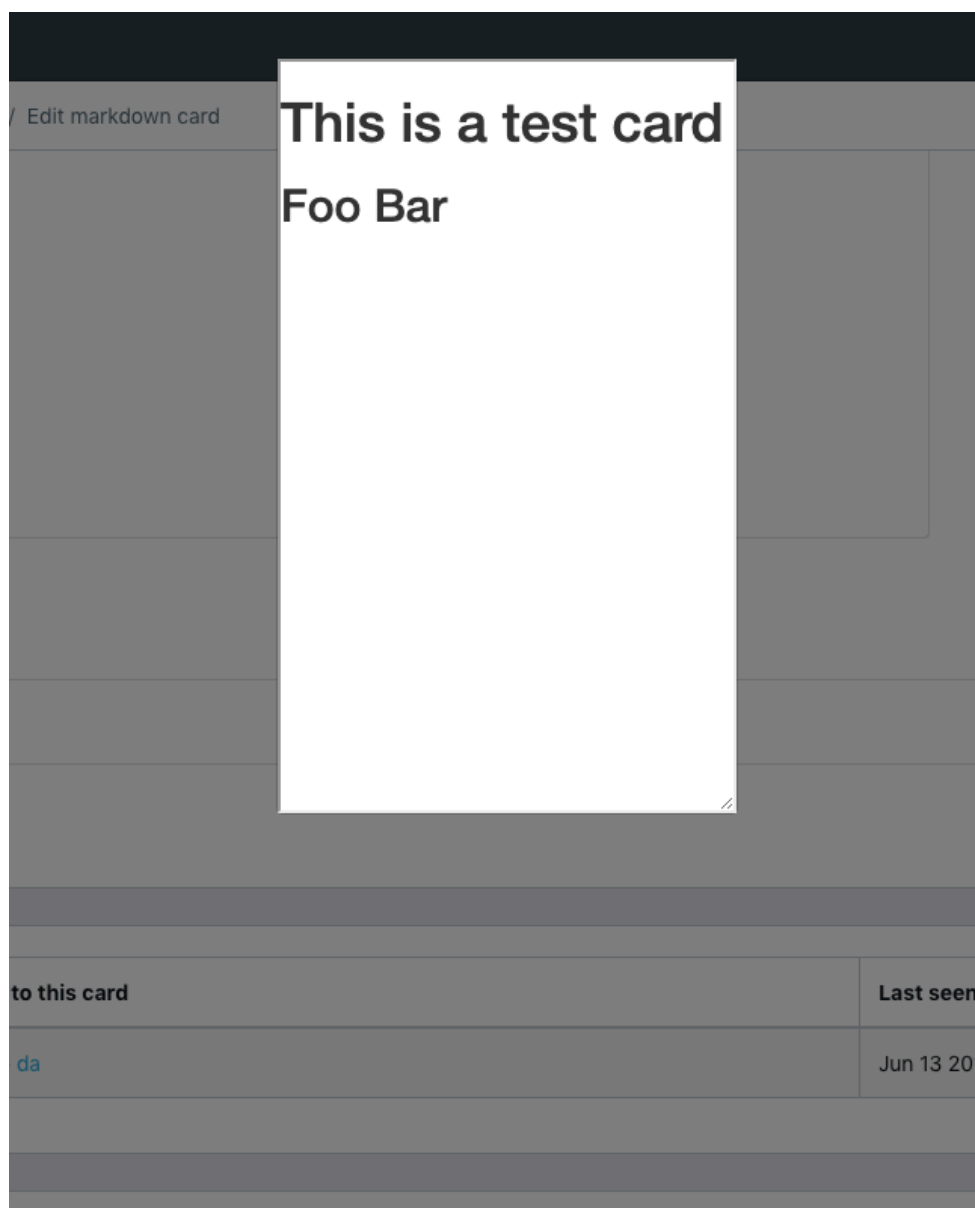
The markdown card edit window

Here one can edit the name and the body of the markdown card. The body can be written in standard markdown syntax like # signifying <h1> tag in html, ## signifying <h2> tag, etc.

There are also options to add images, change the font, add colors, etc. to the markdown card so that it is entirely customisable.

After writing the body of the card, there is a style section where users can select a theme from a preset collection of themes and also enter custom CSS to customise the markdown card.

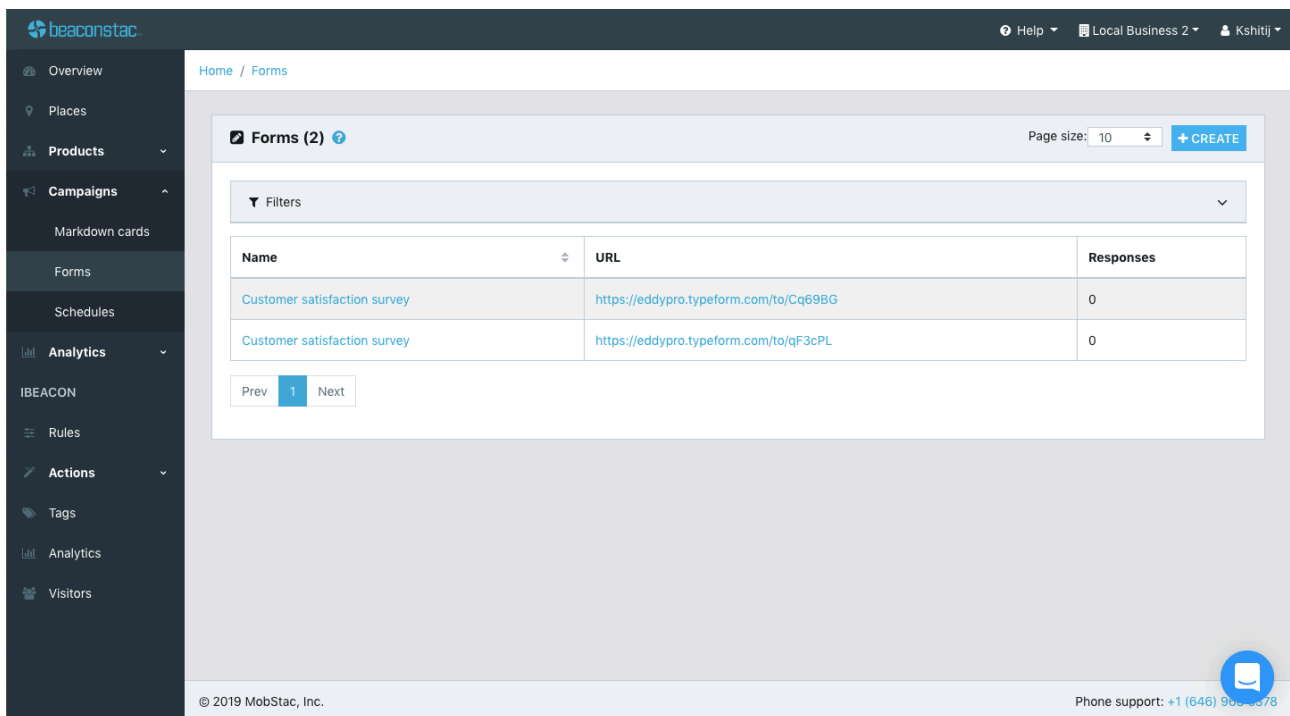
The markdown card created can be previewed in real time in the bar present beside this section.



The markdown card preview

There is also a section which displays the beacons and the rules attached to the markdown card.

2. Forms: The second type of campaign which can be customised is forms.



Name	URL	Responses
Customer satisfaction survey	https://eddypro.typeform.com/to/Cq69BG	0
Customer satisfaction survey	https://eddypro.typeform.com/to/qF3cPL	0

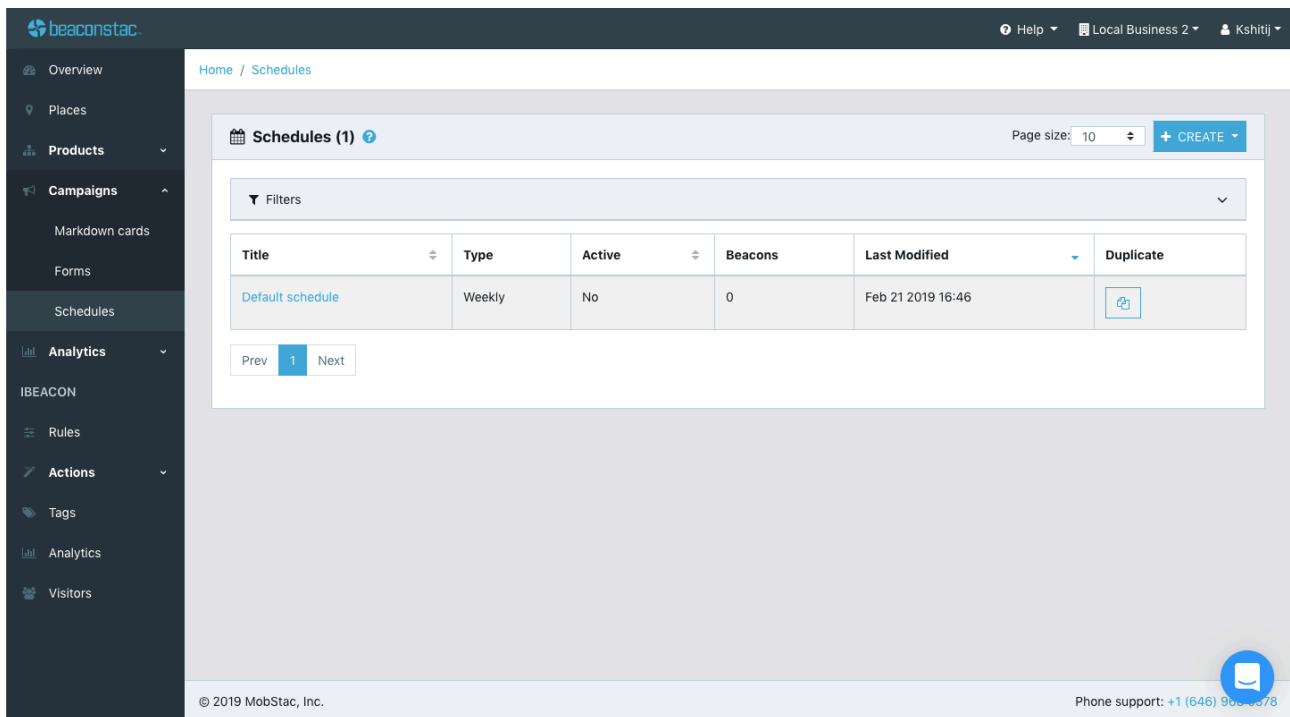
The forms section of the dashboard

The forms section of the dashboard contains details of the forms along with the URL of the form and the number of responses recorded for those forms.

On clicking a form, the following information is visible:

- The read-only forms title and URL.
- The beacons attached to the form

3. Schedules: The third type of campaign is scheduled. Schedules are campaigns which can display different campaign at different times of the day, different days of the week, etc. These conditions can be specified while creating the schedules and can also be modified later.



The schedules section in the dashboard

There are various customisation options while creating schedules.

The first section is the Edit Schedule section:

- In the edit section, the title, timezone, and activation status of the schedule can be modified.

The second section is the rules section:

There can be multiple rules associated with a schedule. The following points must be kept in mind while defining the rules:

Note 1: If the times in your rules overlap, the top-most rule is prioritised.

Note 2: Switching to a rule may be delayed by up to 15 minutes.

The screenshot shows the 'Rule 1' configuration page in the Beaconstac dashboard. The left sidebar contains navigation links: Overview, Places, Products, Campaigns (with sub-links for Markdown cards, Forms, and Schedules), Analytics, IBEACON, Rules, Actions, Tags, Analytics, and Visitors. The main content area is titled 'Rule 1' and includes the following sections:

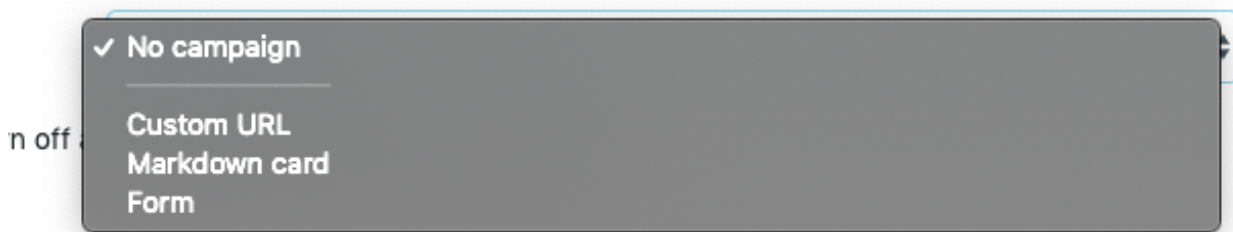
- Days:** A row of buttons for each day of the week: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, and Sunday.
- Time:** Time selection fields showing '09' to '18'.
- All day:** A toggle switch currently set to 'OFF'.
- Campaign:** A section with the heading 'Set the content of your campaign here.' It includes a 'Type' dropdown menu currently set to 'No campaign' and a message: 'You have chosen to turn off all Eddystone notifications in this rule.'
- Notification:** A section with the heading 'Customize the appearance of your notification here.' It includes a 'Language' dropdown set to 'English (default)' with a 'Set default language' checkbox checked, a 'Title*' field with placeholder text 'Your notification title', and a 'Description' field with placeholder text 'Your notification description'.

The rule section of the dashboard

There are various options with which the schedules can be configured while defining rules.

- First, the days of the week in which the schedule will be active can be configured.
- Secondly, the time range in those days when the schedules will run can also be set.

- Also, the campaign which will run in the schedule can be chosen.



- Lastly, the notification appearance of the campaigns can also be configured with options like language, description, and icon.
- There is also a section which displays the beacons attached to the schedule.

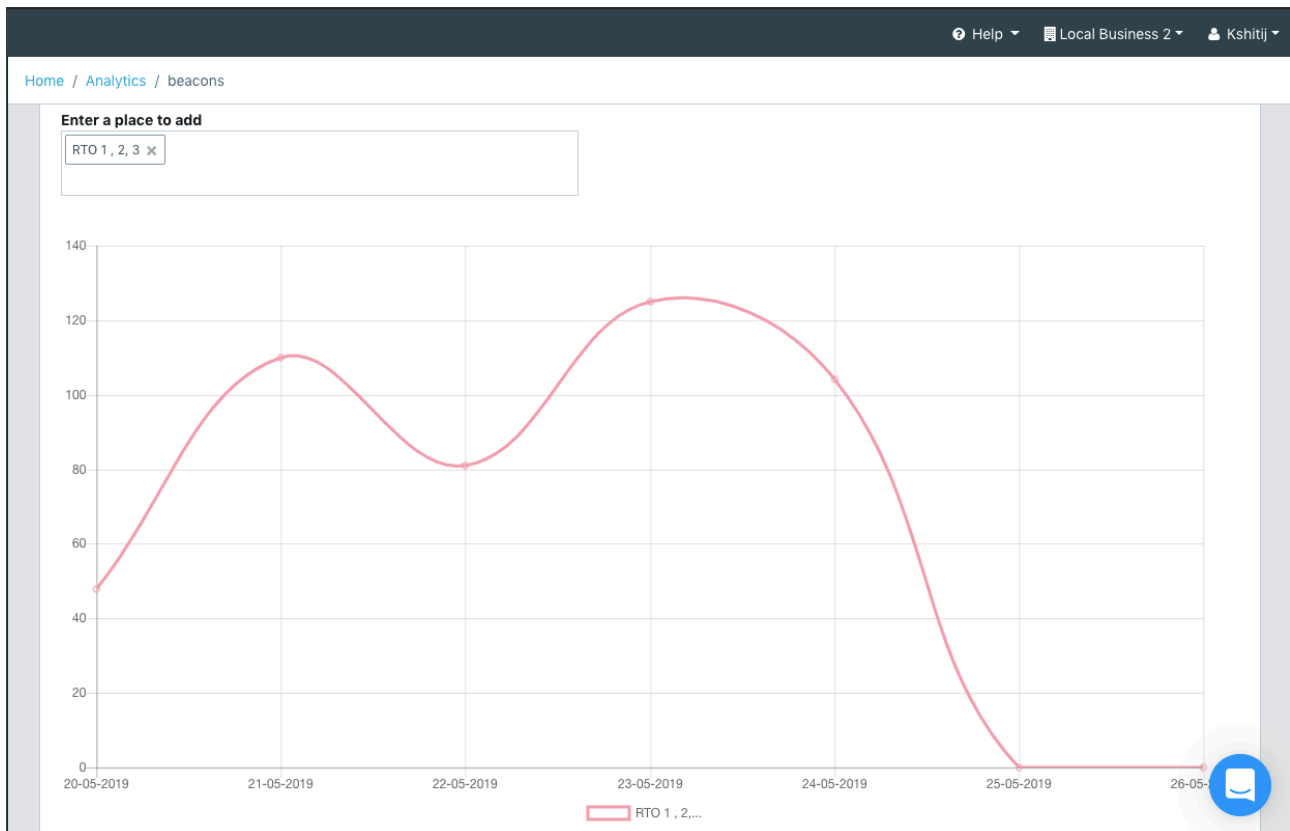
Now let us come to the Analytics section

5. *Analytics*

The analytics section of the dashboard contains analytics data for beacons and NFC tags. In addition to the graphs visible in the analytics sections of beacons and NFC tags subsections, this section also has options to plot the analytics data of different places in the same graph.

This data is available for clicks, CTRs as well as the landing page distribution between campaigns - markdown cards, forms, and links.

The graph data can also be downloaded in XLS format for all the sections in both summary and details form.



Analytics graph for different places

The analytics section also shows details about the top performing beacons/ NFC tags and also the total number of unique visitors.

The data for displaying the graphs and other analytics data is retrieved from the Beaconstac server in real time and is kept in sync so that it can keep track of recent changes in the data.

The number of unique visitors is calculated based on the number of individual devices the campaigns are hit from.

There is a separate section on the bottom left of the navigation bar which is specifically meant for the iBeacon protocol of beacons. This section contains customisations for iBeacon specific features.

6. *iBeacon*

As discussed above, iBeacon is a protocol designed by Apple, which provides some extra set of features not available with Eddystone.

The first among those is 'rules' which is described below:

The screenshot shows a web interface for configuring iBeacon rules. At the top, there is a dark navigation bar with links for 'Help', 'Local Business 2', and a user profile 'Kshitij'. Below this, a breadcrumb trail reads 'Home / Rules / Add rule'. The main content area is titled 'Set up rules that get triggered when the device comes in range of the beacon'. It contains several sections: 'Title*' with a text input 'Test Rule 1'; 'Trigger this rule on' with a dropdown set to 'Entry', followed by 'after 10 seconds.'; an 'Activated' toggle switch currently in the 'ON' position; 'Attach beacons' with a sub-header 'Select one of the options below to attach beacons that would trigger this rule.' and two dropdowns, 'Single beacon' and 'Super 123'; 'Select actions' with a sub-header 'Select a type of action you want your rule to trigger and then select one of the actions pre-defined under that type.'; and two dropdowns for 'Notification' (set to 'None') and 'Webhook' (set to 'Test webhook'). At the bottom left is a link 'Add custom filters'. A blue circular chat icon is visible in the bottom right corner.

The rules section inside the iBeacon header

The rule is a feature which allows beacons to be customised based on the position of the device in a beacon zone starting from the entrance of the device till its exit. The beacons can send customised notifications to the users based on their position inside the beacon zone.

For example, when a person enters the zone, he will be greeted with a welcome message by a beacon deployed near the entrance. Let's consider the beacon zone is a supermarket which has various sections, and there are beacons all around which covers the entire supermarket.

- The iBeacon beacons can be configured to push notifications based on the region where the customer is presently standing.
- It can also be configured in a way that a beacon will only send custom notifications about a particular section of the supermarket when the person is there for a certain period for that specific section.
- The rules can be set to trigger the beacon after a given time from the entry/exit of a person in a particular zone for which the rule is set.
- On being triggered the notifications to send can be customised and webhooks can be attached to those notifications once the user clicks the notifications.
- A **webhook** in web development is a method of augmenting or altering the behaviour of a web page, or web application, with custom callbacks. These callbacks may be maintained, modified, and managed by third-party users and developers who may not necessarily be affiliated with the originating website or application.
- These web hooks can be used by the third party users to get notifications about important events when and where they occur.

- While defining web hooks, various custom filters can be added to trigger the callback only when those rules are satisfied.

Next is the *Actions* section:

The actions section contains the notifications and webhooks which are used in defining rules of iBeacon beacons.

Notifications can be created for both Android and iOS devices, and the following things can be configured while creating a notification:

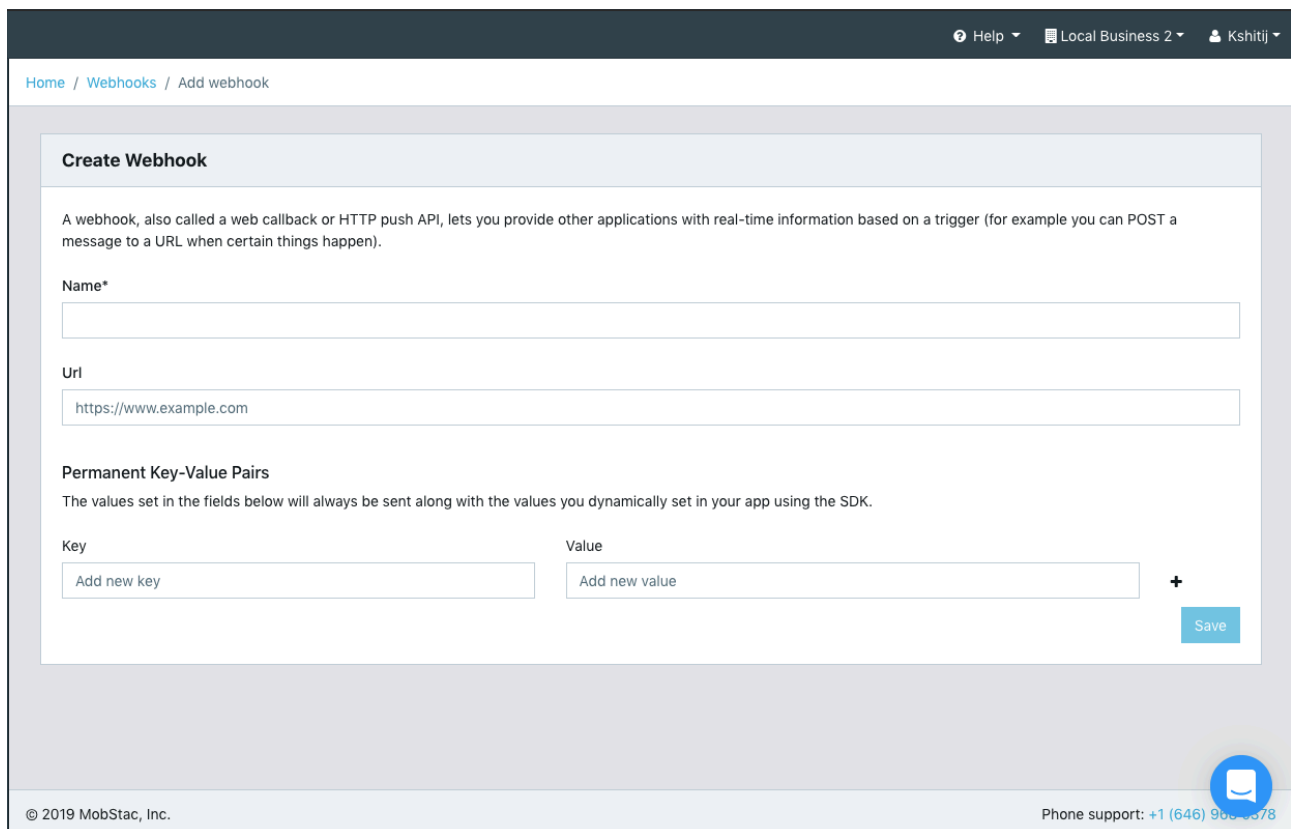
Title, message, the image inside the notification, and the type of CTA(Call to Action) which can be chosen between markdown cards and URL.

The screenshot displays the 'Create Notification' interface. At the top, there's a navigation bar with 'Home / Notifications / Create notification' and a user profile 'Kshitij'. The main heading is 'Create Notification'. Below it, a descriptive text states: 'Local notifications enable an app that isn't running in the foreground to let its users know it has information for them. Unlike a push notification, a local notification is scheduled and sent by the app itself.' The form includes a 'Title*' field, a 'Message*' field, and a 'Call to action' section. The 'Call to action' section has a 'Type' dropdown set to 'Markdown card' and a text input field containing 'random_card'. A 'Drag and drop' area with an upload icon is provided for a promo image. To the right, there's a preview of a smartphone showing a notification with the title 'Your Title Appears Here' and the message 'Your message appears here. Use this space.' The phone's status bar shows 'TATA DOCOMO' and '15:00'. A 'Locked' status indicator is visible above the phone. A blue chat bubble icon is in the bottom right corner.

The create notification screen inside the Actions section

For webhooks, the following things can be configured:

The name, URL and the permanent key-value pairs which will be sent along with the callback whenever the webhook is triggered.



The screenshot shows the 'Create Webhook' screen within a web application. At the top, there is a dark navigation bar with links for 'Help', 'Local Business 2', and a user profile 'Kshitij'. Below this, a breadcrumb trail reads 'Home / Webhooks / Add webhook'. The main content area is titled 'Create Webhook' and includes a descriptive paragraph: 'A webhook, also called a web callback or HTTP push API, lets you provide other applications with real-time information based on a trigger (for example you can POST a message to a URL when certain things happen)'. The form contains three main sections: 1. 'Name*' with a text input field. 2. 'Url' with a text input field containing 'https://www.example.com'. 3. 'Permanent Key-Value Pairs' with a sub-header explaining that values set here will be sent with SDK values. This section has two input fields labeled 'Key' (containing 'Add new key') and 'Value' (containing 'Add new value'), separated by a '+' sign. A 'Save' button is located at the bottom right of the form. The footer of the page includes the copyright '© 2019 MobStac, Inc.' and a phone support number '+1 (646) 966-3378' next to a chat icon.

The create webhooks screen inside the Actions section

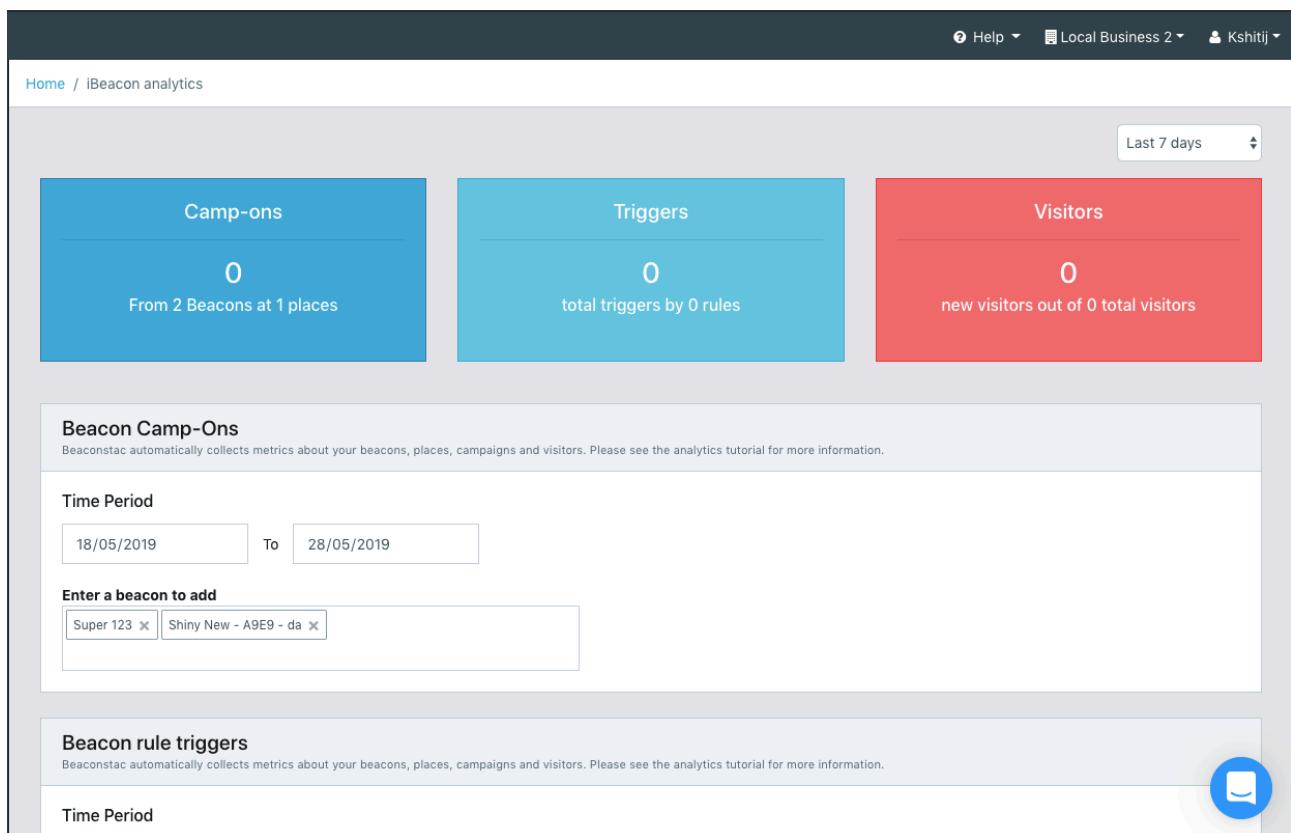
Other values can also be sent dynamically along with these key-value pairs which need to be configured from the app of the iBeacon customer using the Beaconstac SDK.

One essential requirement to use the iBeacon functionality is that the customer must have a native app into which the iBeacon functionality can be activated using the Beaconstac SDK, i.e., it is not an app-less solution.

Next section is the *Tags* section:

- Tags are created to group beacons, which can be used to attach rules to a group of beacons in a single shot.
- Tags can be configured with a name, and beacons can be attached to a particular tag.
- Whenever any changes are made to the rules for a tag, it will automatically cascade into all the beacons under that tag.
- In addition to that, the beacons can be configured individually as well.

Next section is the *iBeacon Analytics* section:



The iBeacon analytics screen

- The iBeacon analytics section gives information about the following things:
- Camp-ons, triggers, and visitors.
- Camp-ons are the number of notifications from the iBeacon beacons which are clicked by the users.
- Triggers correspond to the rules corresponding to the iBeacon beacons. It shows the number of rules triggered by the iBeacon beacons,
- The visitor's section shows the number of visitors coming in the range of the iBeacon beacons. It gives information about both unique and total visitors.
- All these properties can be viewed for different time periods.

The last section is a separate *Visitors* section for iBeacon customers:

This section displays in details the data from visitors. This includes the following:

- The device name, the last location of the visitor in the iBeacon range, which is one of the places assigned to the beacons and the device to which the notifications were sent.
- This can also be viewed for different time periods, like other analytics data.

Now let us come to the Account Control section:

This section displays as well as provides options to the Beaconstac customers to set/modify certain account settings depending upon the privileges he/she has.

For example, a master organisation can control and view all the info from all the child organisations whereas a child organisation can only view the details of his/her account.

Home / Account

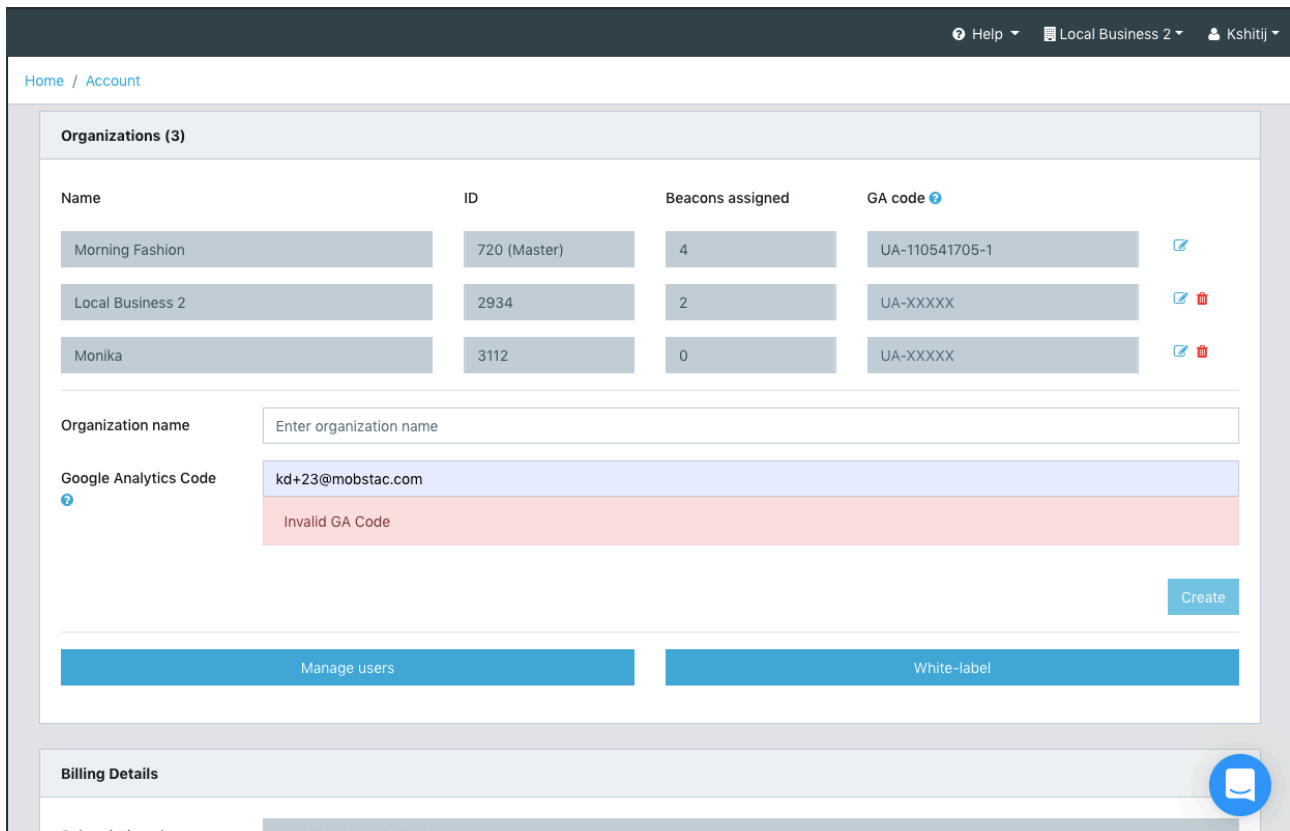
Account Details

First name*	Kshitij
Last name*	Deo
Email	kd+23@mobstac.com
Account type	Reseller
Total Beacons	6
Organization	Morning Fashion (720)
Date joined	Mon Jul 13 2015
Developer Token	e62435a78e67ec98bba3b879ba00448650032557
Physical Web for Android	<input checked="" type="checkbox"/>
Wallet Support	<input checked="" type="checkbox"/>

The accounts section in the dashboard

The info which is displayed is basic account related things like name, email, account type – one of Basic, Premium, Reseller, and White-label, the total number of beacons assigned to the account, the org name and the developer token.

If it is a master organisation the account section also provides options for creating child organisations as well as editing/deleting the existing child accounts, which are done through the manage users section.



The child organisation management section

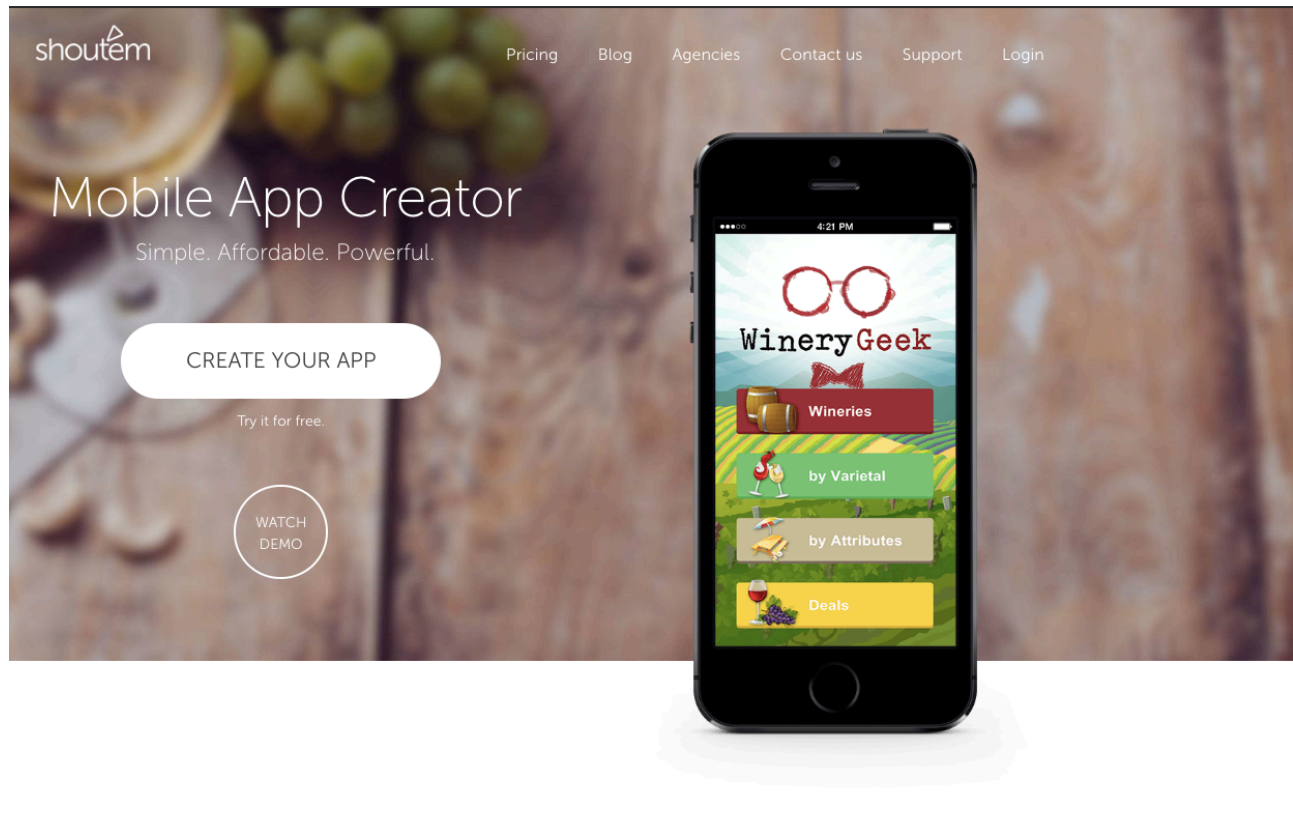
The next sections is a read-only section which displays information regarding the subscription plan, the next billing date, and the payment method, which can be changed.

This section also provides options to change the plan type of customer.

It also has a reset password section for changing the password of the account.

The dashboard also has an app builder section for customers who are subscribed to it. Clicking on this section redirects the user to the **Beaconstac Shoutem** app builder page.

- Shoutem is an excellent platform for building apps for non-developers, which doesn't require coding and provides a lot of customisations.
- Shoutem supports app development for both iOS and Android devices



A glimpse of the Shoutem app builder

With this, I come to the end of the Angular dashboard design section.

Abiding by company privacy policy, **I cannot share any code.**

Next, let us move on to the solution of the Chatbot.

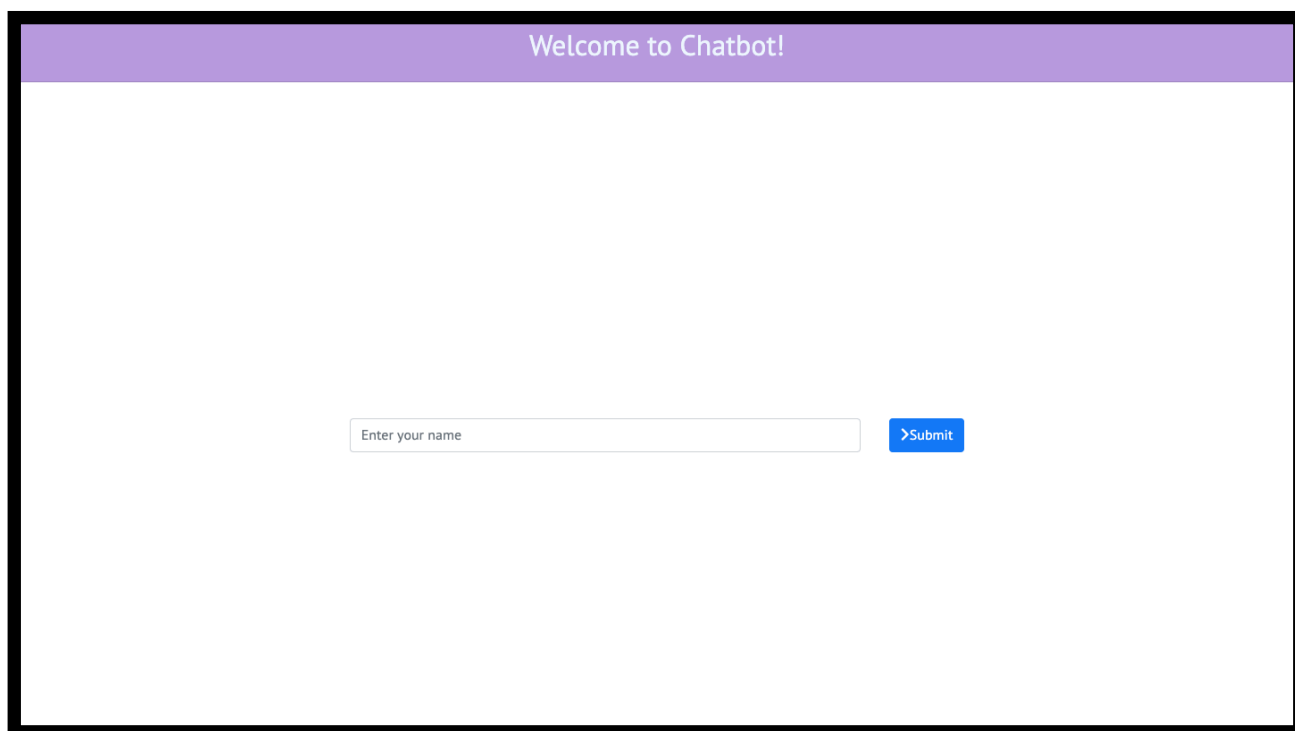
Chatbot:

After choosing the right technology stack as described in the requirements section, the next step was to implement the bot, keeping in mind all the conditions the bot should satisfy.

- Initially, I focused on building the server for our bot, which is the most important module contributing to the functionality of the bot.
- The server is built on the Django framework. Besides implementing the basic server configurations like routing, template rendering, I have also implemented the bot response module in Django.
- For the user interface, I have chosen VueJS as our driver framework along with Bootstrap 4 for building a beautiful UI for the end users.

Now let us go through the basic flow of the app, which I have tried to keep very simple for rich user experience.

- A. Initially, when the user opens the bot, it displays a login screen where we take the name of the user as input, which is used to make the bot personalised for the user.
- B. Once the user submits his/her name, he/she lands in the main bot window. The window displays Welcome [USERNAME] at the top and greets the user with a set of categories of FAQs to choose from. I have tried to include all possible common categories which users usually look for.

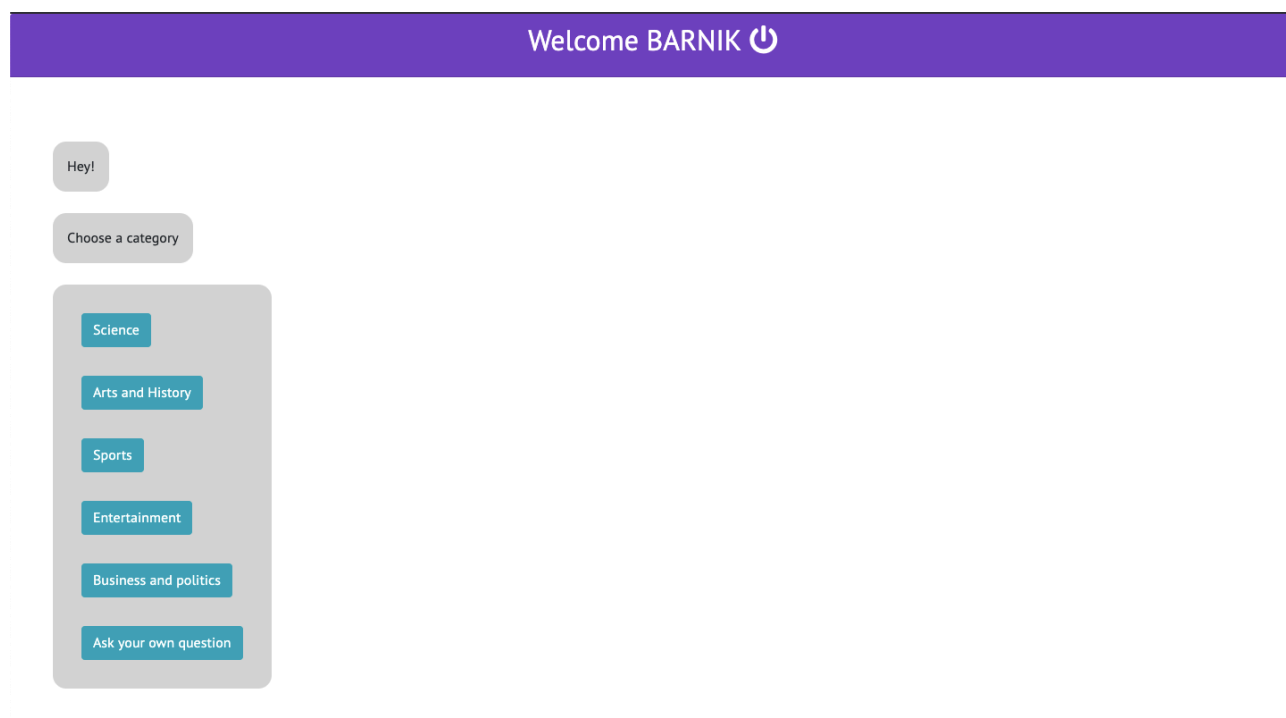


Welcome to Chatbot!

Enter your name

>Submit

The opening screen of the bot



Welcome BARNIK

Hey!

Choose a category

Science

Arts and History

Sports

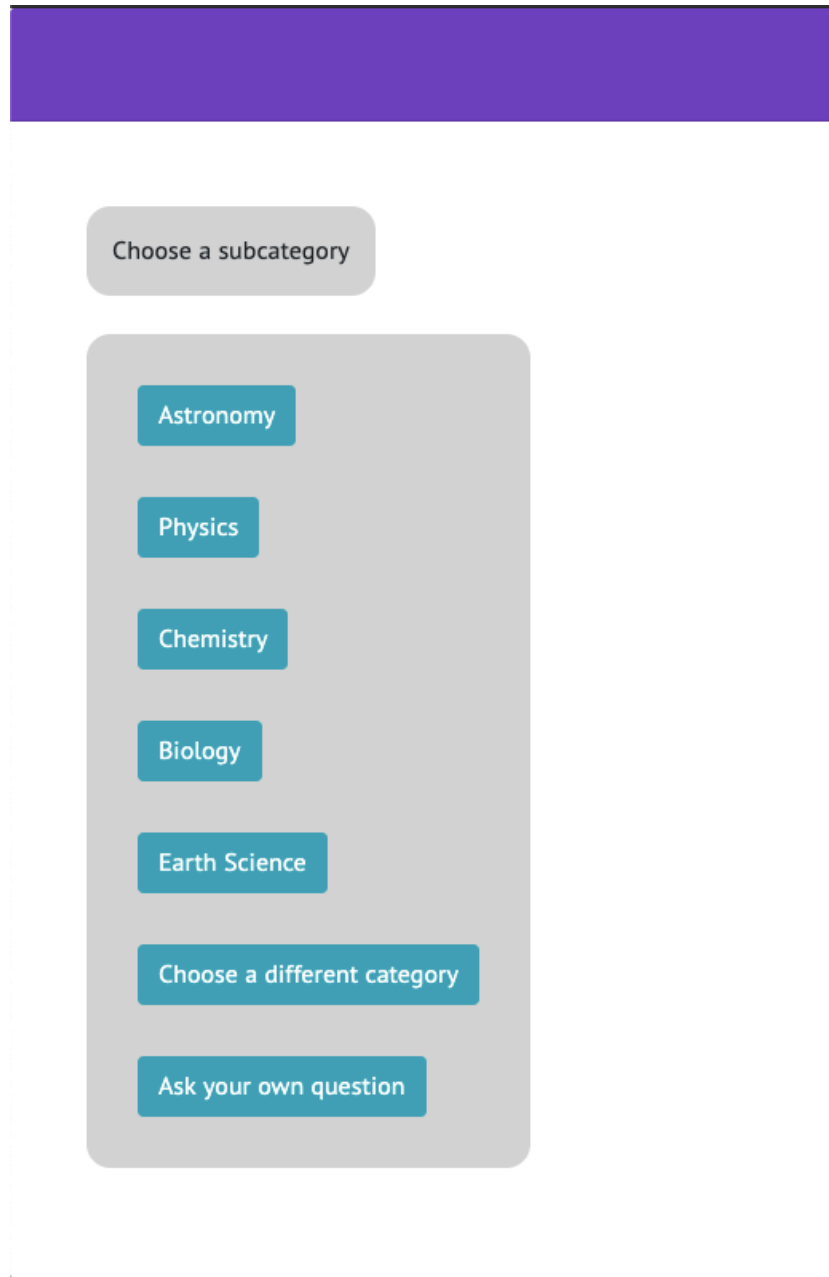
Entertainment

Business and politics

Ask your own question

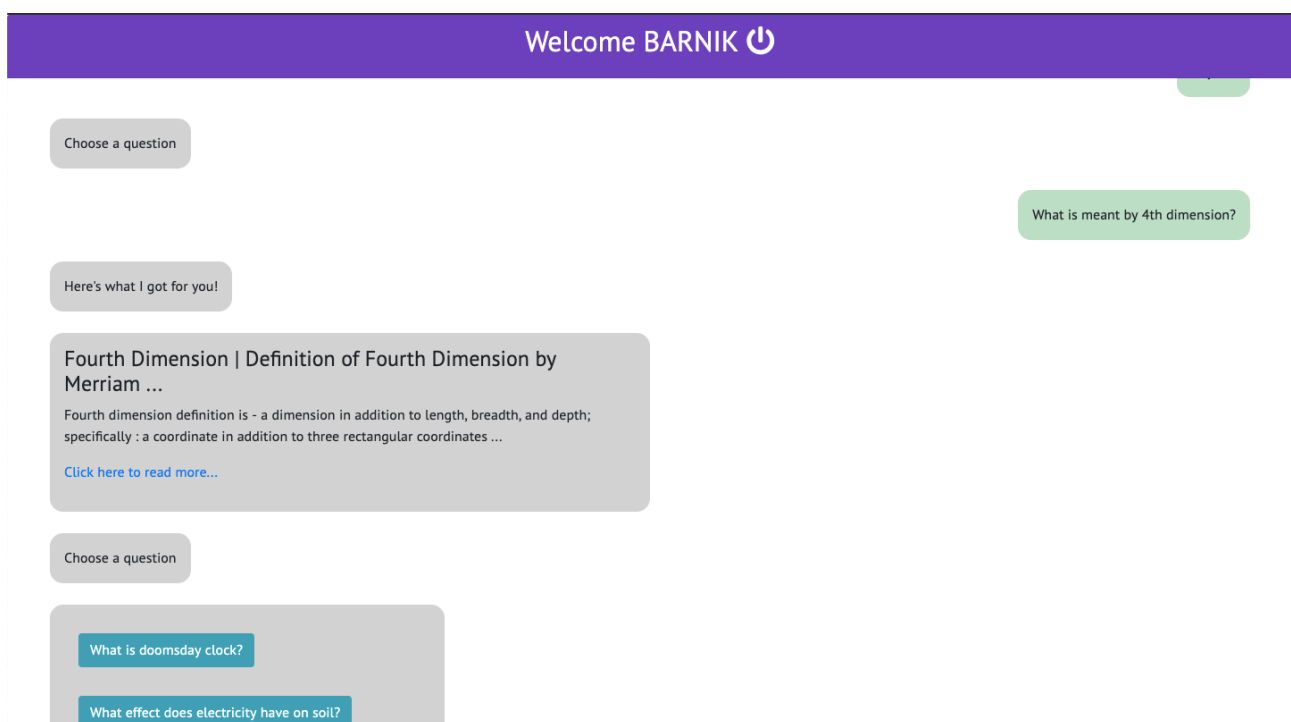
The welcome screen after the user logs in

- C. On selecting a category, the user is presented with a list of subcategories relevant to the chosen category.



List of categories shown by the bot

- D. On selecting a subcategory, we present the user with a set of commonly asked and interesting questions belonging to the chosen subcategory.
- E. When the user selects a question, the bot responds with an answer from the server. In the background, we send an XHR(XML Http Request) to the server along with the question, and the server implements the following algorithm:



Bot response when the user chooses a question

- It accepts the request, and after retrieving the question, it sends it to the RASA server.
- RASA then processes the data, and the RASA NLU operates on the data, which then extracts the necessary intents and the entities from the question.

- Let us now understand what do we mean by intent and entity.
- An intent can be thought of the basic category of the question. For example, if the user sends 'Hello,' the intent can be thought of 'greeting.' Similarly, 'Good morning,' 'Hey! How are you doing?' are also under the intent 'greeting.'
- An entity is a set of characteristics of the intent which help RASA to further understand the demand of the question so that an accurate answer can be provided for the same.

For example, the user asks the question: 'What is the weather now in Bangalore?'

- Here the intent of the question is 'weather,' and the entity corresponding to the 'weather' intent is the 'location' for which the user wants to know the weather. Thus, here, Bangalore is the 'location' entity.
- After processing this question, the RASA NLU will respond with {'intent'; 'weather'; 'entity'; 'Bangalore'}.
- To make the RASA bot understand about the intents and their corresponding entities we need to train the NLU with the same.
- To train RASA, there is a trainer.json file which needs to be filled with training examples, and specifies the intent and entities for those examples.

A snippet from the trainer.json file is given file:

```
{
  "text": "bye",
  "intent": "goodbye",
  "entities": []
},
{
  "text": "Who is the president of India?",
  "intent": "general_faq",
  "entities": [
    {
      "start": 11,
      "end": 20,
      "value": "president",
      "entity": "designation"
    },
    {
      "start": 24,
      "end": 29,
      "value": "India",
      "entity": "country"
    }
  ]
},
{
  "text": "who is the prime minister of England?",
  "intent": "general_faq",
  "entities": [
    {
      "start": 29,
      "end": 36,
      "value": "England",
      "entity": "country"
    }
  ]
}
```

```

    },
    {
      "start": 11,
      "end": 25,
      "value": "prime minister",
      "entity": "designation"
    }
  ]
}

```

- In addition to the values of the entities, the position of the entity is also specified in terms of the starting and the ending character.
- Now we need to train the NLU with the given data and after training the NLU will be ready to answer questions corresponding to the trained intents and entities.
- After training the NLU with the intents and entities, we have to train the dialogue management model. This is important because the bot needs to understand what the user is looking for and if any info is missing from the user's question, the bot can ask for the same. This makes the bot a lot more interactive.
- For training the dialogue management model, we have to create a stories.md file which contents actual conversations between the user and the bot in intent and entity format.
- Before we can move on to the stories.md file we have to specify the actions the bot will take when certain entities are encountered. This has to be specified in a separate yml file.

A snippet from the yml file is given below:

```
templates:
  utter_greet:
    - 'Hello, how can I help you?'
    - 'Hi, I am here to help.'
  utter_goodbye:
    - 'Bye :('
  utter_ask_country:
    - 'What country?'
```

- The actions can be templates as can be seen above or they can be custom actions for which data needs to be retrieved from the server.
- After defining the domain of the bot, we move on to the stories.md file.

A snippet from the stories.md file is given below:

```
## Story 1
* greet
  - utter_greet
* general_faq
  - utter_ask_country
* general_faq{"country":"India"}
  - utter_goodbye
```

- Here we can define whether an entity is required in intent so that the bot has the complete information before retrieving the answer.
- In the above example if the user only asks 'Who is the prime minister?' Then the bot will respond with 'What country?'. After the user responds to the same, the answer will be retrieved.
- RASA also provides functions for training the dialogue management model online, i.e. while having an actual conversation with the bot. These conservations can then be automatically stored into the stories.md file by RASA.
- After the dialogue management model is trained, one thing we need to take care of is custom actions.
- For custom actions, I have used the google custom search engine which I have fine-tuned and customized for answering FAQs.
- The Google custom search engine API responds with a set of best matching answers for the question.

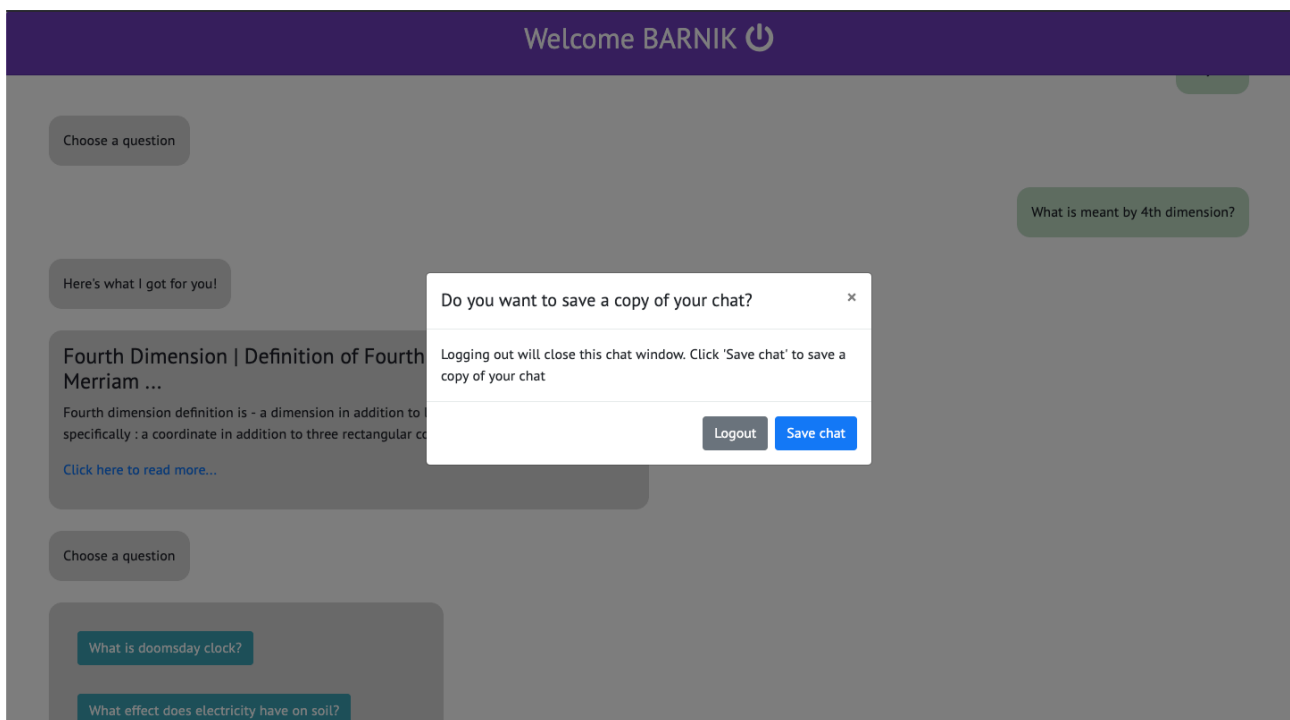
The code snippet for the custom search call is given below:

```
def fetchResponse(question):  
    service = build("customsearch", "v1", developerKey=API_KEY)  
    res = service.cse().list(  
        q=question,  
        cx=CSE_ID,  
        num=5  
    ).execute()
```

```
res = res['items'][0]
response = {'link': res['link'], 'title': res['title'], 'desc': res['snippet']}
print(question)
print(response)
return response
```

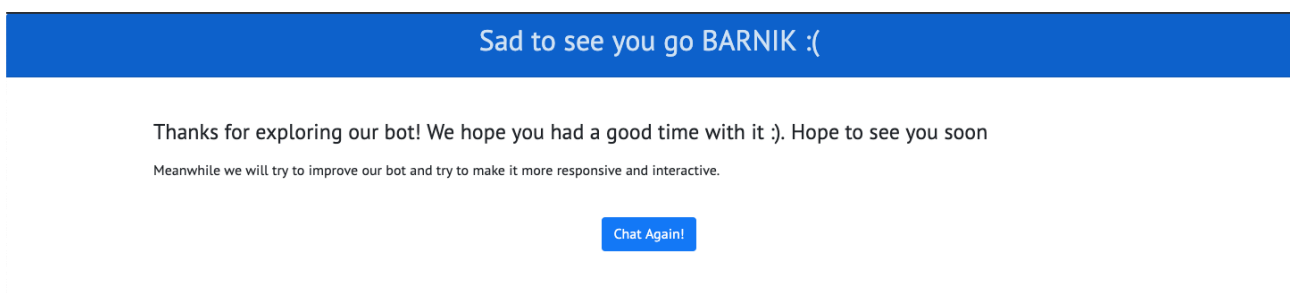
Here we are sending a request to the Google CSE.

- After it responds, we then take the top response from the set of answers and send it to the XHR promise after formatting the answer.
- After every answer, the bot provides there is a 'Read More' link which links to the website from which the response has been retrieved.
- In any stage, I have provided options for the user to jump to a different category, subcategory, or question to improve the user experience.
- I have also integrated a 'Ask your question button', which opens up a form where the user can type in his/her question. We retrieve the answer to their questions in the same way as we do for the set of preset questions to keep parity in the method we use for both cases.
- Anytime the user wants to quit the session, he/she can click/tap on the logout icon beside the username which opens up a modal with the following options:
 - Save Chat - I thought the users would like to save a copy of their conversation with the bot and hence I have kept a 'Save chat' option for the same. It downloads the entire conversation from the beginning in a .txt file.
 - Logout - I have also included a logout option which logs the user out of the present session.



The logout modal

- On logging out, the user is presented with a 'Thank You' screen along with an option to start a new session afresh.



The logout screen of the bot

Given below is a flowchart which shows the entire process flow of the bot:

Retrieving and processing the question

The server accepts the request, and after retrieving the question, it sends it to a Google custom search engine, which we have fine-tuned and customized for answering FAQs.



Retrieving answer

The Google custom search engine API responds with a set of best matching answers for the question.



Ranking and formatting

Then, we take the top response from the set of answers and send it to the XHR (XML Http Request) promise after formatting the answer.



Displaying answer

After every answer, the bot provides a 'Read More' link, which links to the website from which the answer has been retrieved.

With this, I come to the end of the design of the bot.

Next, I would like to list a few possible improvements which I plan to implement on the bot to improve its performance further by making it more responsive and interactive.

- I plan to integrate more diversity in the responses, which are sent from the bot, including media files like images, which makes it more intuitive for certain types of questions.
- Once the above features are implemented successfully, I will try to integrate voice support into the bot.

Validation Results:

Angular Dashboard

For the Angular dashboard, I have attached screenshots of the working dashboard in the Solution/Methodology section.

Django Chatbot

For the chatbot, the working bot is deployed in Heroku and can be accessed via the following URL:

<https://faq-bot-django.herokuapp.com>

I have also attached relevant screenshots of the bot in the Solution/Methodology section.

References:

Angular Dashboard

<https://www.beaconstac.com/proximity-marketing>

<https://www.beaconstac.com/what-is-a-bluetooth-beacon>

<https://www.beaconstac.com/nfc-marketing>

<https://www.beaconstac.com/qr-code-marketing>

<https://angular.io/>

Django Chatbot

<https://docs.djangoproject.com/en/2.2/>

<https://vuejs.org/v2/guide/>

<https://getbootstrap.com/docs/4.3/getting-started/introduction/>

<http://rasa.com/docs/getting-started/>

<https://developers.google.com/custom-search/v1/introduction>