

#BullhornEngage

ENGAGE

How to Get the Best MoBull Experience

Tanvi Gadre
Jaclyn Moore
Amrutha Rajiv



Tanvi Gadre
Development Manager



Jaclyn Moore
Software Engineer



Amrutha Rajiv
Principal Software
Architect

ENGAGE

Applying the Law of Two Feet

Every individual has two feet, and must be prepared to use them.

Individuals can make a difference and must make a difference. If that is not true in a given situation, they, and they alone, must take responsibility to use their two feet, and move to a new place where they can make a difference.

-Harrison Owen

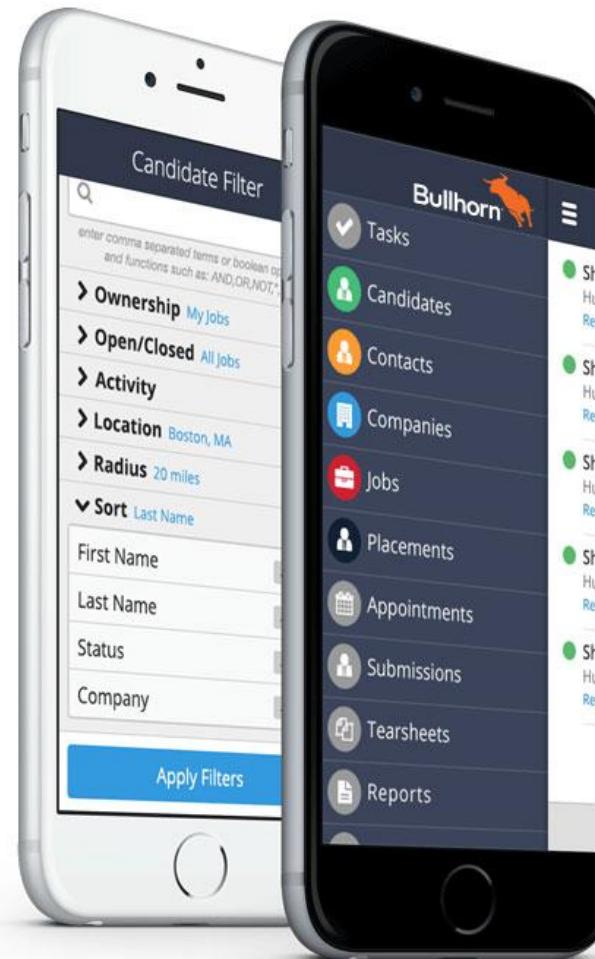
What Will We Cover Today?

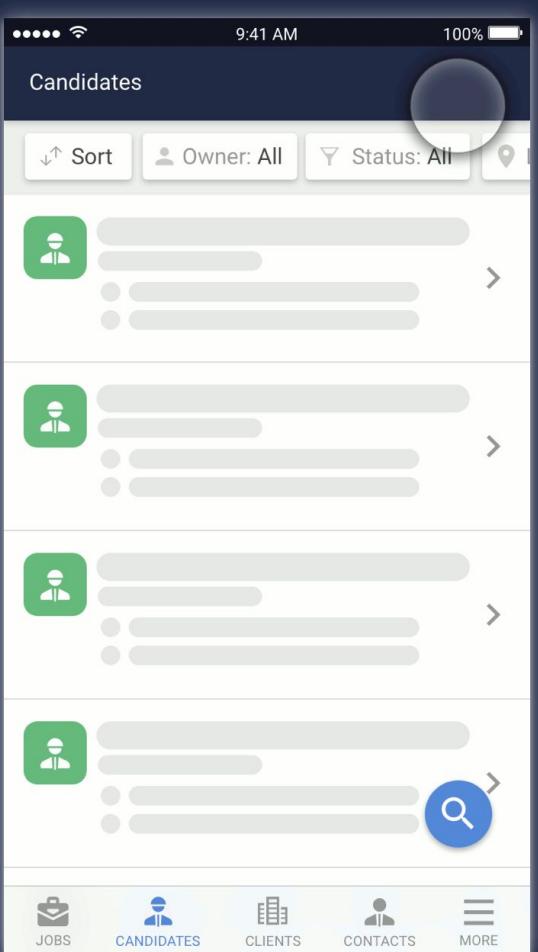
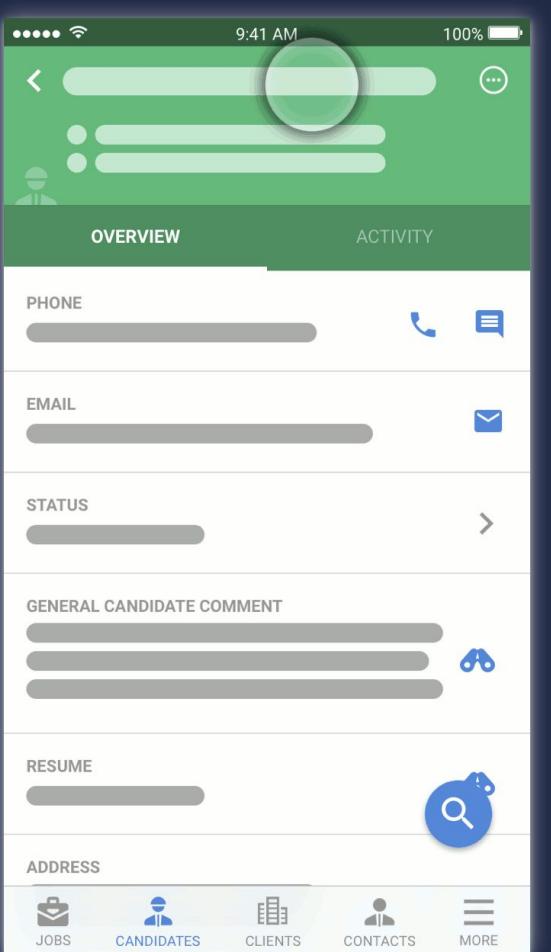
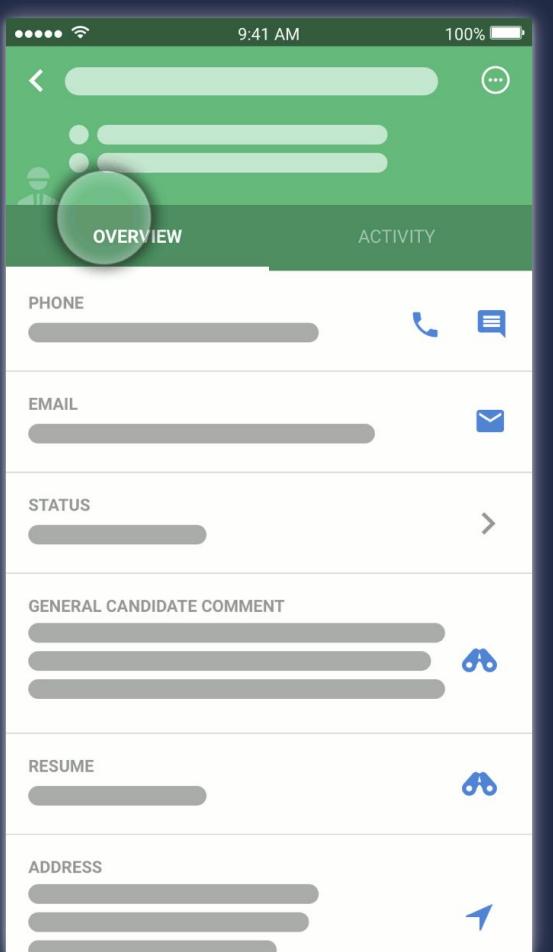
- Mobile Overview
- Custom Actions
- Field Interactions
- Page Interactions
- Running of MoBull

Bullhorn Mobile Tech Overview

ENGAGE

m.bullhorn







Live Demo



What is the Stack?

What is the Stack?

- Ionic
- Angular
- Ionic Angular
- Bullhorn
 - Novo-Elements
 - Ionic Mocks
 - Taurus
 - Chomsky

Novo Design System

#BullhornEngage

ENGAGE

Novo & Mobile

The image shows a laptop screen and a smartphone screen side-by-side, illustrating the Bullhorn software's desktop and mobile versions.

Laptop Screen (Bullhorn Software Interface):

The laptop displays the Bullhorn software interface. At the top, there is a navigation bar with the Bullhorn logo, a search bar, and buttons for "Find" and "Add". On the right side of the top bar, there are links for "Privacy", "Help", and a user profile for "Emily Swartz". Below the top bar is a secondary header with "Client Contacts" and a search bar. The main area is a "Contact List" table with columns: Id, Name, Company, Title, and Status. The table contains 18 rows of contact information. At the bottom of the table are buttons for "Parse Resume" and "Close All Tabs".

	Id	Name	Company	Title	Status
9728	Caroline Felts	AT&T Center	Owner	Welcome Letter	
9731	Karen Goldstein	240 & Western	Owner	Welcome Letter	
9757	Kelly Park	Altamesa and McCart	Owner	Welcome Letter	
9579	Mike Robinson	Coveris Corporate- Chicago, IL	Vice President	Vice President	
9583	Edwin Tsang	Coveris- Ontario, CA	HR Manager	Vice President	
9840	Stacy Coolidge	Servpro of Edmond	Executive VP		
674	Sabina Ramakrishnan	Marriott International	Regional Manager		
10110	Ginna Raahauge	Catholic Health Initiatives	Chief Technology Of...		
10111	Glenn R. Yarbrough	Catholic Health Initiatives	Director, IT Strategi...		
10801	Brenda Wallace	Mercy General Hospital - Sacramento	Nurse Manager		
10863	House Warehouse	House	House Contact for W...		
5840	Tom Tiger	BP Oil	Director, FBA Techn...		
4892	Gina Meyers	Apple, Inc.	Call Center Manager		
4907	Harvey Hodge	Heinz	Call Center Manager		
4908	Elizabeth Heideman	Johnson & Johnson	Call Center Manager		
4910	Myrtle Buchanan	MTD Products	Call Center Manager		

Smartphone Screen (Mobile App Interface):

The smartphone displays the mobile application interface. At the top, it shows the time (9:41 AM) and battery level (100%). The main screen shows a contact card for "Nathan Whitney" from "Mass General Hospital". The card includes sections for "OVERVIEW", "ACTIVITY", "CONTACT NAME" (Nathan), "POSITION", "COMPANY" (Mass General Hospital), "MOBILE PHONE", "EMAIL" (n.whitney@gmail.com), "DIRECT PHONE", and "ADDRESS". Below the card are tabs for "CANDIDATES", "CONTACTS", "CLIENTS", "JOBS", and "MORE".

#BullhornEngage

ENGAGE



What is **ionic**?

What is Ionic?

- Powerful HTML5 SDK using web technologies
- Targeted at building Hybrid Mobile app
- Provides broad range of native mobile components, animations and UI
- iOS and Android



How is it Native?

How is it Native?

- Ionic Native is a TypeScript wrapper for Cordova/PhoneGap plugins that make adding any native functionality

A close-up photograph of two hands holding a dark-colored smartphone. The hands are positioned vertically, with the thumbs on the left side of the phone. The background is blurred, showing what appears to be a person's shirt and a red strap.

Live Demo

So what does this
mean for you?



ENGAGE

Custom Actions

ENGAGE

What Can They Do?

- Perform custom actions
- Actions like
 - Invoke a standard list with prefilled params
 - Pre fill an external form with Candidate/Job information e.g Adding background screening info
 - Display custom interface like Mass Placement Termination

How Do You Configure Them?

The screenshot shows a configuration interface for a Field Map Entity named "Candidate". The "Custom Menu Actions" tab is active. A modal window is open, allowing the creation of a new custom menu action. The modal fields include:

- Name: * Add Background screening
- Enabled:
- Location: On a record's action list, on the Mobile app
- URL: (empty input field)
- Partner Name: (empty dropdown menu)

Buttons in the modal include "Save" (blue), "Delete" (red X), and "Move" (up and down arrows).

How Do They Work?

- They appear on the Actions on a record or on a list
- Will have to set up a separate action for mobile but use the same URL
- Open the url on click on a new page

Custom Actions Demo

```
34     self.log.setLevel(logging.INFO)
35     self.debug = debug
36     self.logger = logging.getLogger(__name__)
37     if path:
38         self.file = open(os.path.join(path, "request.log"), "w")
39         self.file.seek(0)
40         self.fingerprints.update(self._read())
41
42     @abstractmethod
43     def from_settings(settings):
44         debug = settings.getbool("supersecret.debug")
45         return cls(job_dir(settings), debug)
46
47     def request_seen(self, request):
48         fp = self.request_fingerprint(request)
49         if fp in self.fingerprints:
50             return True
51         self.fingerprints.add(fp)
52         if self.file:
53             self.file.write(fp + os.linesep)
```

Field Interactions

ENGAGE

What Can They Do?

- Scripts that can customize form fields
- Examples:
 - Update picker options for a customText field to bring back data from an external system
 - Filter down options on a contact dropdown based on a selection made on a different field
 - Mark a field as invalid and disallow users from being able to save the formdata

How Do You Configure Them?

Field Map Interactions:

Entity: * Candidate

Private Label: * BH-62056 - S-Enabled Engineering-Only SL9 Test Corporation

Column Name: * customText1

Display: * Novo SL 9

Event: * On Change

Run on Init:

Is Enabled:

Sort Order: *

Script: *

Cancel Save Delete

BHAS: sl9app3:81:BULLHORN_LG1901:BULLHORN4086: 853

Alternatively, use
bullhorn-cli.

Field Interactions API

- Example API Object

```
{  
    currentEntity: "Candidate",  
    currentEntityId: 5874,  
    currentKey: "status",  
    http: {...},  
    appBridge: {...},  
    globals: {  
        entitlements: {...},  
        settings: {...},  
        user: {...}  
    },  
    isAdd: false,  
    isEdit: true,  
    modalService: {...},  
    toastService: {...}  
}
```

What is Possible?

- getActiveKey
- getActiveValue
- getControl
- markAsInvalid
- modifyPickerConfig
- markAsDirty
- displayToast
- disable

Page Interactions

ENGAGE

What Can They Do?

- Scripts that can customize display of fields, actions and activity sections
- Examples:
 - Hide the “Add submission action” from your recruiters when the candidate dateAvailable has not been filled out
 - Update display of customText fields to include a custom ID and Label
 - Update the label of the Notes section on the activity tab to read as “Candidate Comments”

How Do You Configure Them?

The screenshot shows the Novo UI interface for managing page interactions. On the left, there's a sidebar with fields for NAME (set to 'Record'), PAGE (set to 'Record'), ACTION (dropdown), SORT ORDER (set to 1), and an ENABLED checkbox which is checked. Below these is a code editor containing a function definition:

```
function (API) {
```

On the right, a modal window titled "Page Interaction Documentation" provides information about the API object and its properties. It includes a code snippet for logging the API object:

```
console.log('API', API);
```

It also lists several properties of the API object:

```
API = {
  globals: {
    settings: Array,
    entitlements: Object,
    user: {
      corporationId: number,
      corporationName,
      email: string,
      name: string,
      firstName: string,
      lastName: string,
      locale: string,
      masterUserId: number,
      privateLabelId: number,
      swimlaneId: number,
      userId: number,
      userType: number,
      username: string
    }
  },
  appBridge: AppBridge,
  toastService: NovoToastService,
  modalService: NovoModalService,
  currentEntity: string,
  currentEntityTrack: string,
  currentEntityId: string,
  pageContext: string
}
```

Alternatively,
use
bullhorn-cli.

What is Possible?

- Modify Actions
 - Change Label
 - Mark Disabled
 - Mark Hidden
- Modify Overview Fields
 - Change Label
 - Change Display Data
 - Mark Disabled
 - Mark Hidden
 - Convert ID Field To Data
 - Convert ID Field To Data w/ Link
- Modify Activity Sections
 - Mark Hidden
 - Update Style
 - Custom Class Name
- Add Edit Presave
 - Mark Form as Invalid
 - Show a Warning Prompt
 - Display Toast

Page Interactions API

- Example API Object

```
{  
    currentEntity: "Candidate",  
    currentEntityId: 5874,  
    http: {...},  
    appBridge: {...},  
    globals: {  
        entitlements: {...},  
        settings: {...},  
        user: {...}  
    },  
    pageContext: "Record",  
    modalService: {...},  
    toastService: {...}  
}
```

Novo vs. Mobile

- You can use the same interactions across mobile and novo
- API will give you information about the platform it's running on - if it's desktop or mobile, It will also give you other settings pertaining to the logged in user

Bullhorn CLI & Configuration Starter

- Using the cli to install page & field interactions via command line
- Clone configuration starter to manage your page and field interactions



[bullhorn/bullhorn-cli](https://github.com/bullhorn/bullhorn-cli)

[bullhorn/extension-configuration-starter](https://github.com/bullhorn/extension-configuration-starter)

Page & Field Interactions Demo

```
34     self.log.setLevel(logging.INFO)
35     self.debug = debug
36     self.logger = logging.getLogger(__name__)
37     if path:
38         self.file = open(os.path.join(path, "request.log"), "w")
39         self.file.seek(0)
40         self.fingerprints.update(self.request)
41
42     def from_settings(settings):
43         debug = settings.getbool("BULLHORN_DEBUG")
44         return cls(settings_dir(settings), debug)
45
46     def request_seen(self, request):
47         fp = self.request_fingerprint(request)
48         if fp in self.fingerprints:
49             return True
50         self.fingerprints.add(fp)
51         if self.file:
52             self.file.write(fp + os.linesep)
53
54     def request_fingerprint(self, request):
```

Running of the MoBull



ENGAGE

How Do I Build It?

- Git clone repo
- Npm install
- Ionic serve



[bullhorn/mobullDemo](https://github.com/bullhorn/mobullDemo)

What About on a Device?

- Android Studio
- Xcode - mac only



[Ionic V3 Documentation](#)



#BullhornEngage

ENGAGE

A photograph showing two people from behind, both holding white smartphones. They are looking at their screens, which display various app icons. The background is dark and out of focus.

Questions?