ENGAGE

Extensions - Your Bullhorn Superpower

Amrutha Rajiv
Brian Kimball
Joshua Godi

Brian Kimball
ENTERPRISE ARCHITECT

Joshua Godi
PRINCIPAL UI MEOWNGINEER

Amrutha Rajiv
SOFTWARE ARCHITECT

ENGAGE

What we will cover today

- BOTs
- Field Interactions
- Page Interactions
- Novo-Elements Updates
- Bullhorn CLI

Bots

ENGAGE

What can they do?

- Update associated records
  - When a job is lost update submission status to "Lost to Competition"
  - On end of placement (and no other active placements exist) update candidate status and date available
- Configure custom workflows
  - If placement pay rate or bill rate was updated, update corresponding record in ADP or Peoplesoft.

How do they Work?

- Bots are executed on save
- Data updates done via bots can be seen on the Edit History of records. User who does the updates is the Bots User.
- Bots can be disabled or deleted

How do you configure a bot?

- Add a Bot via the REST API
- Bots can have the following properties
    - Name
    - Conditions - When do you want the bot to execute?
    - Outcomes - What do you want the Bot to do?

```
const closeJob: Bot = {
  name: 'Job Closed',
  botType: 'POSTSAVE',
  isEnabled: true,
  onUpdate: true,
  entity: 'JobOrder',
  botConditionData: {
    botConditions: [
      {
        entity: 'JobOrder',
        field: 'status',
        matchOperator: 'EQUALS',
        value: 'Closed',
      },
    ],
    logicOperator: 'AND',
  },
  botOutcomes: [
    {
      outcomeType: 'UPDATEENTITY',
      data: {
        isOpen: false,
      },
      entity: 'JobOrder',
    },
  ]
```

When a job status is updated to "Closed" update isOpen field to "false"

```
const placementApproved: Bot = {
  name: 'Placement Approved',
  botType: 'POSTSAVE',
  isEnabled: true,
  onUpdate: true,
  entity: 'Placement',
  botConditionData: {
    botConditions: [
      {
        entity: 'Placement',
        field: 'status',
        matchOperator: 'EQUALS',
        value: 'Approved',
      },
    ],
    logicOperator: 'AND',
  },
  botOutcomes: [
    {
      outcomeType: 'UPDATEENTITY',
      entity: 'Candidate',
      data: {
        status: 'Placed',
      },
    },
```

When a placement status is updated to “Approved” update Candidate status to “Placed”

```
const placementDateEnd: Bot = {
  name: 'Placement dateEnd',
  botType: 'POSTSAVE',
  isEnabled: true,
  onUpdate: true,
  entity: 'Placement',
  botConditionData: {
    botConditions: [
      {
        entity: 'Placement',
        field: 'dateEnd',
        matchOperator: 'ISUPDATED',
      },
    ],
    logicOperator: 'AND',
  },
  botOutcomes: [
    {
      outcomeType: 'UPDATEENTITY',
      entity: 'Candidate',
      data: {
        dateAvailable: '$dateEnd+1',
      },
    },
```

When a placement dateEnd is updated, update candidate dateAvailable to dateEnd + 1.

```
  botType: 'POSTSAVE',
  isEnabled: true,
  onUpdate: true,
  entity: 'Placement',
  botConditionData: {
    botConditions: [
      {
        entity: 'Placement',
        field: 'payRate',
        matchOperator: 'ISUPDATED',
      },
      {
        entity: 'Placement',
        field: 'billRate',
        matchOperator: 'ISUPDATED',
      },
    ],
    logicOperator: 'OR',
  },
  botOutcomes: [
    {
      outcomeType: 'UPDATEENTITY',
      entity: 'Candidate',
      data: {
        hourlyRate: '$payRate',
      },
    },
    {
      outcomeType: 'WEBHOOK',
      url: 'https://us-central1-engage-shifts.cloudfunctions.net/updateADP',
      httpMethod: 'POST',
      queryParams: {
        placementID: '$id',
        candidateID: '$candidate.id',
        payRate: '$payRate',
        billRate: '$billRate',
      },
```

If placement pay rate or bill rate was updated, update corresponding record in ADP or Peoplesoft.

```
const placementApproved: Bot = {
  name: 'Placement Approved',
  botType: 'POSTSAVE',
  isEnabled: true,
  onUpdate: true,
  entity: 'Placement',
  botConditionData: {
    botConditions: [
      {
        entity: 'Placement',
        field: 'status',      Brian Kimball, 13 days ago • feat(Bots): Added s
        matchOperator: 'EQUALS',
        value: 'Approved',
      },
    ],
    logicOperator: 'AND',
  },
  botOutcomes: [
    {⋯
    },
    {
      outcomeType: 'WEBHOOK',
      url: 'https://us-central1-engage-shifts.cloudfunctions.net/sendToSlack',
      httpMethod: 'POST',
      queryParams: {
        entity: 'Placement',
        entityId: '$id',
        status: '$status',
        candidate: '$candidate.name',
        job: '$jobOrder.title',
        startDate: '$dateBegin',
      },
    },
  ],
```

If a placement is approved, send me a slack notification!

Building a webhook

```javascript
const { SLACK_WEBHOOK_URL } = require('./config.json');
const { IncomingWebhook } = require('@slack/client');

const webhook = new IncomingWebhook(SLACK_WEBHOOK_URL)

// subscribe is the main function called by Cloud Functions.
exports.sendToSlack = (req, res) => {
  return Promise.resolve()
    .then(() => {
      if (req.method !== 'POST') {
        const error = new Error('Only POST requests are accepted');
        error.code = 405;
        throw error;
      }

      // Send message to Slack.
      const message = createSlackMessage(req.query);
      return webhook.send(message, (result) => {
        console.warn('Sent', result);
        return res.status(200).send('SUCCESS!');
      });
    }).catch((err) => {
      console.error(err);
      return res.status(err.code || 500).send(err);
    });
}
```

Tasks, Emails and more..

- Add tasks and send emails using bots
  - When a job is lost update submission status to "Lost to Competition". **Also, send an email to the job owner that the job was lost.**
  - When a placement dateEnd is updated, update candidate dateAvailable to dateEnd + 1. **Also, add a task for the recruiter with a due date closer to the candidate's dateAvailable to place the candidate in a new job.**
- A single App to rule them all
  - Build a single webhook application to handle requests for all your clients. Session relevant to the client can be created using auth code passed into the webhook by the bot

Resources

- bullhorn/extension-configuration-starter
- bullhorn/bot-samples

What would your bot do?

Field Interactions

ENGAGE


Nothing new… wah wah

Recap

- Exist on all Add/Edit Forms
- Allow you to modify fields based on user input
- Hide/Show/Add fields into any form!
- Documentation inside NovoElements
  - http://bullhorn.github.io/novo-elements/#/field-interactions

Page Interactions

ENGAGE

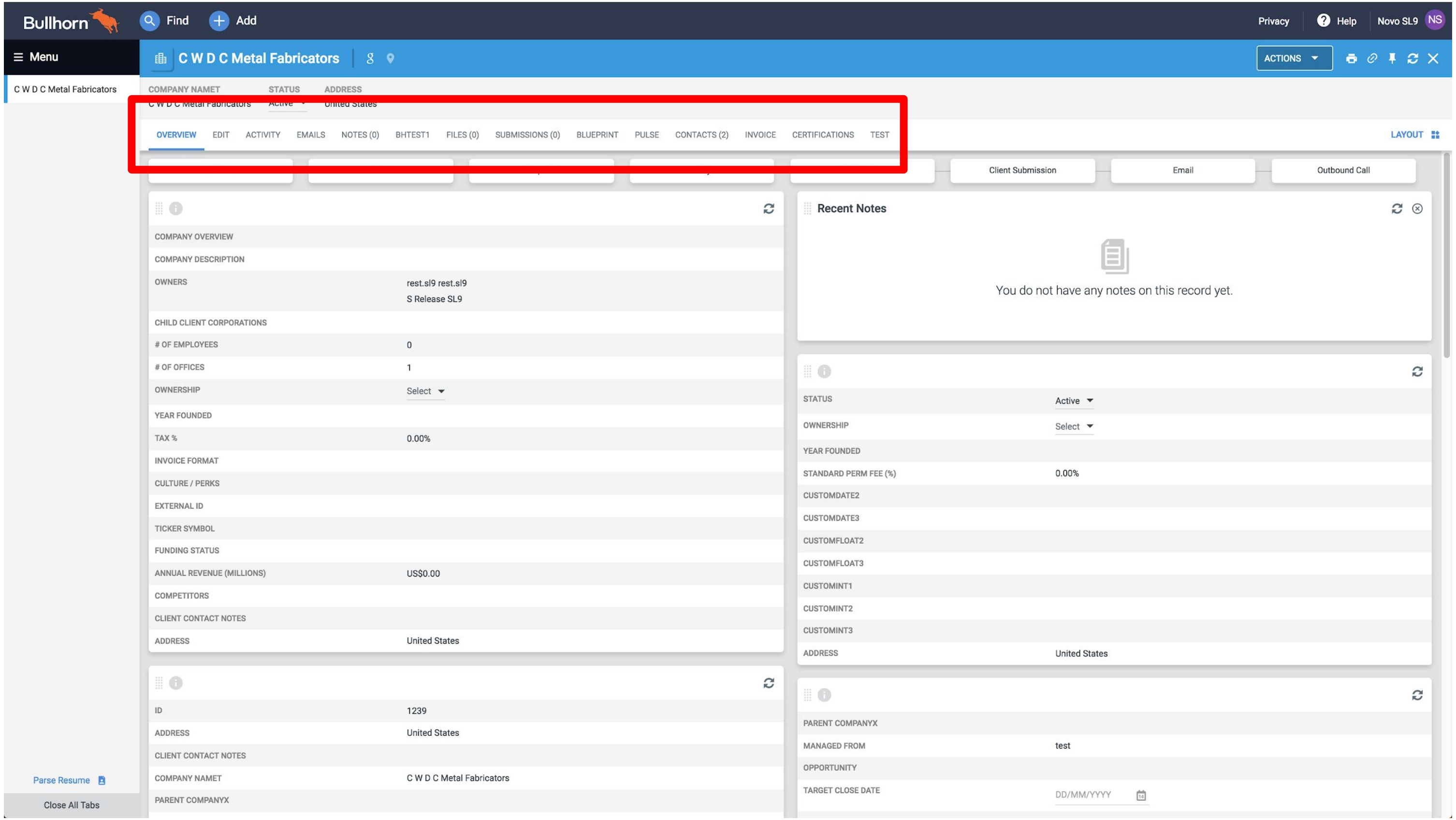
What can they do?

modifying your Bullhorn experience

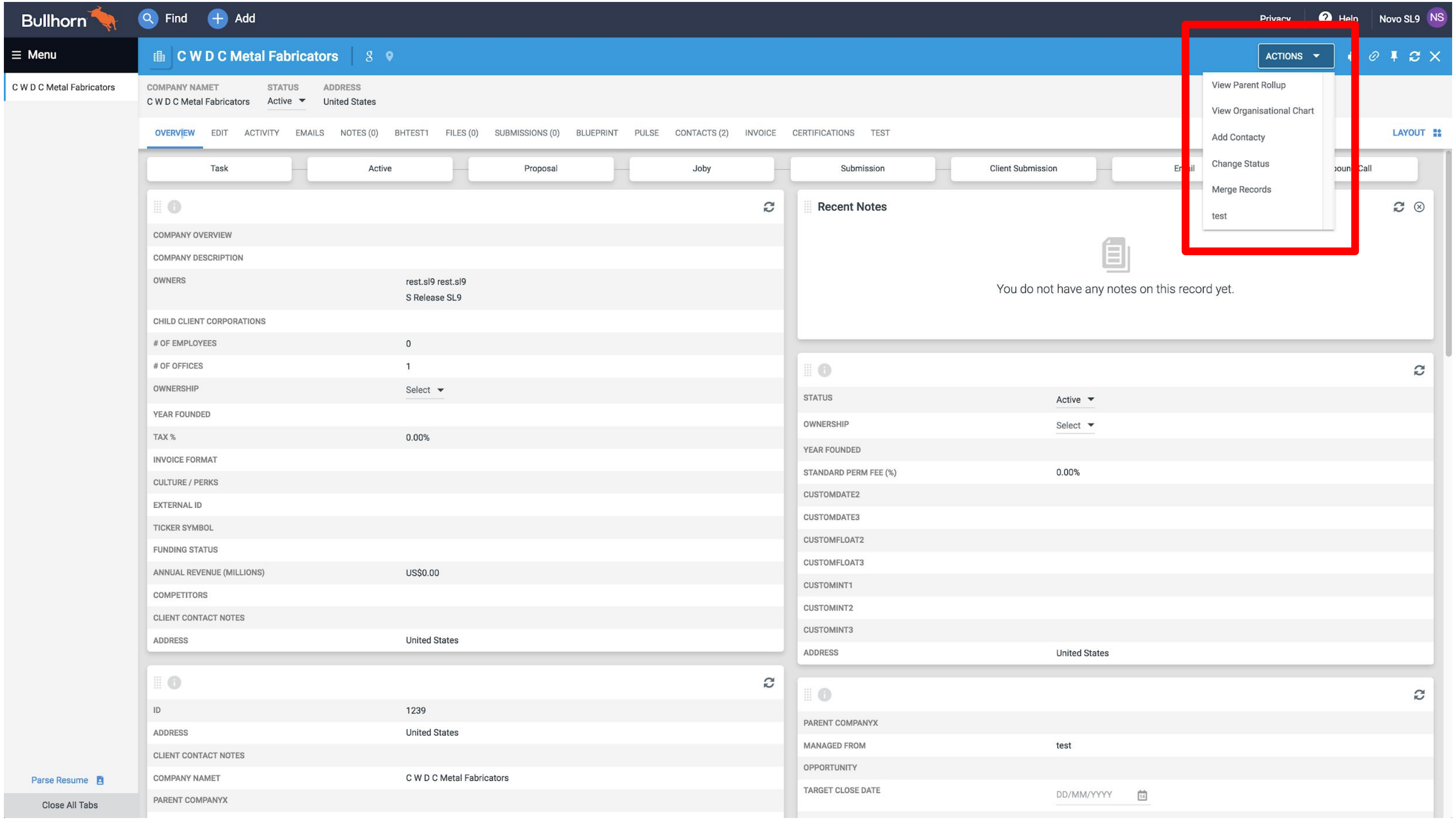
- Modify Record tabs
- Modify Record actions
- Modify Overview fields
- Pre/post saving on add/edit forms
- Customize workflow icons
- More to come!
    - Fast Find / Fast Add / etc...

What can they modify?

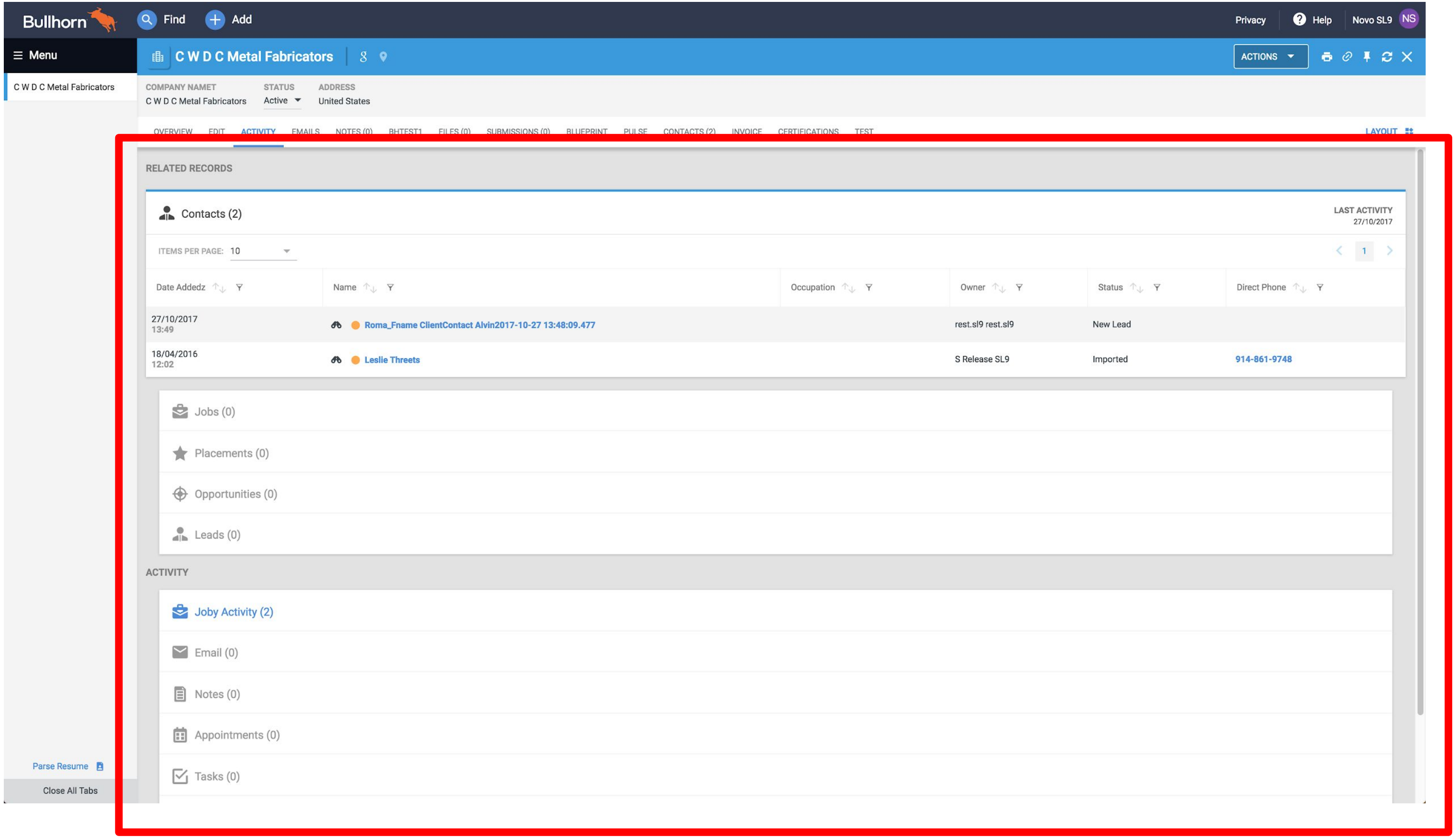
Record Tabs label/style/visibility/disable



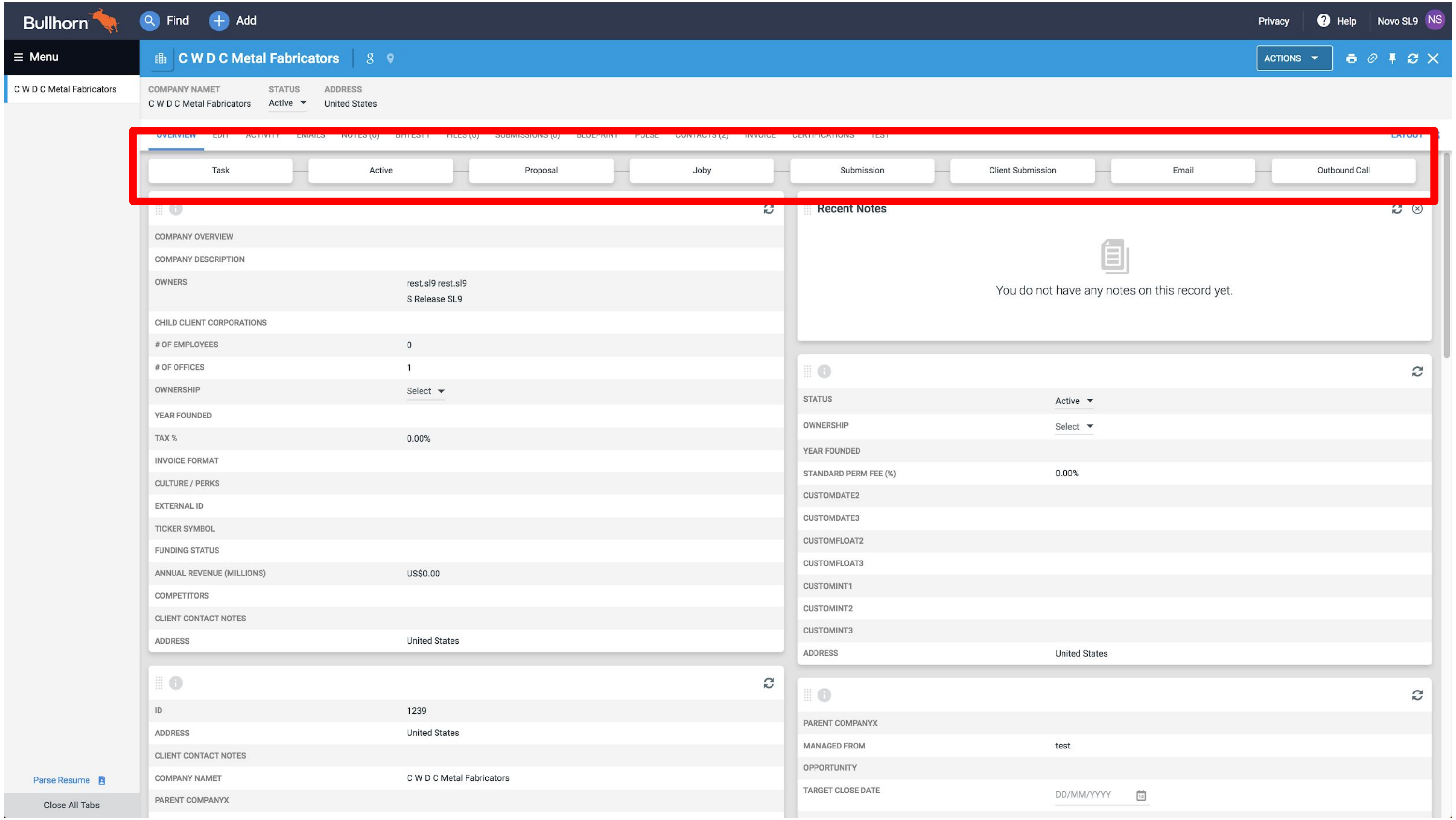
Record Actions label/style/visibility/disable



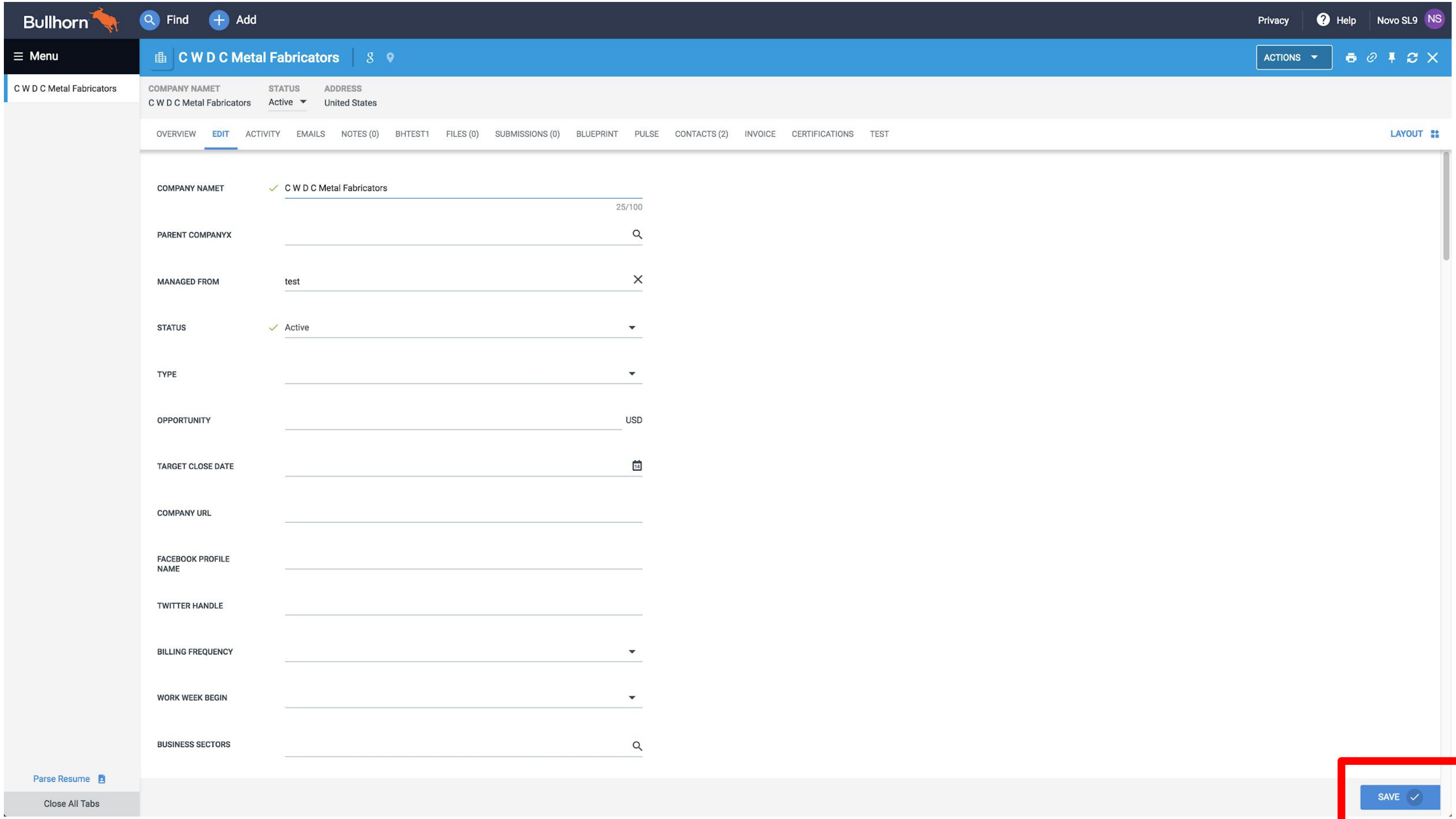
Activity Sections visibility/style

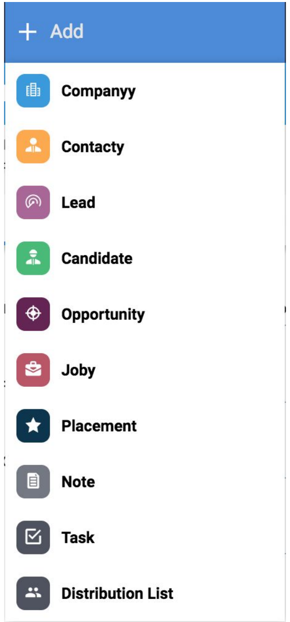


Overview Fields label/value/visibility/style



Add/Edit Form Pre/Post-Save validation/toast/redirect



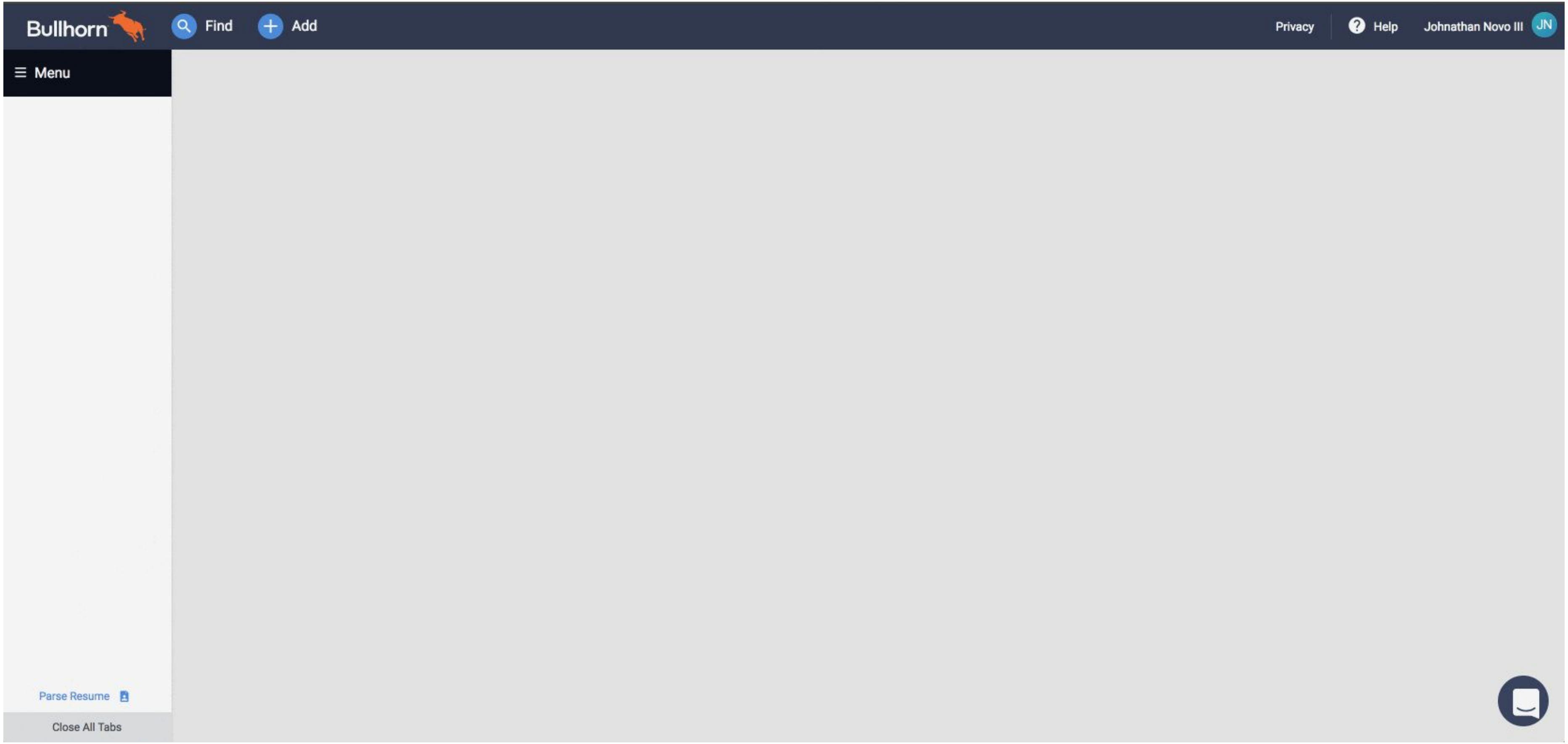


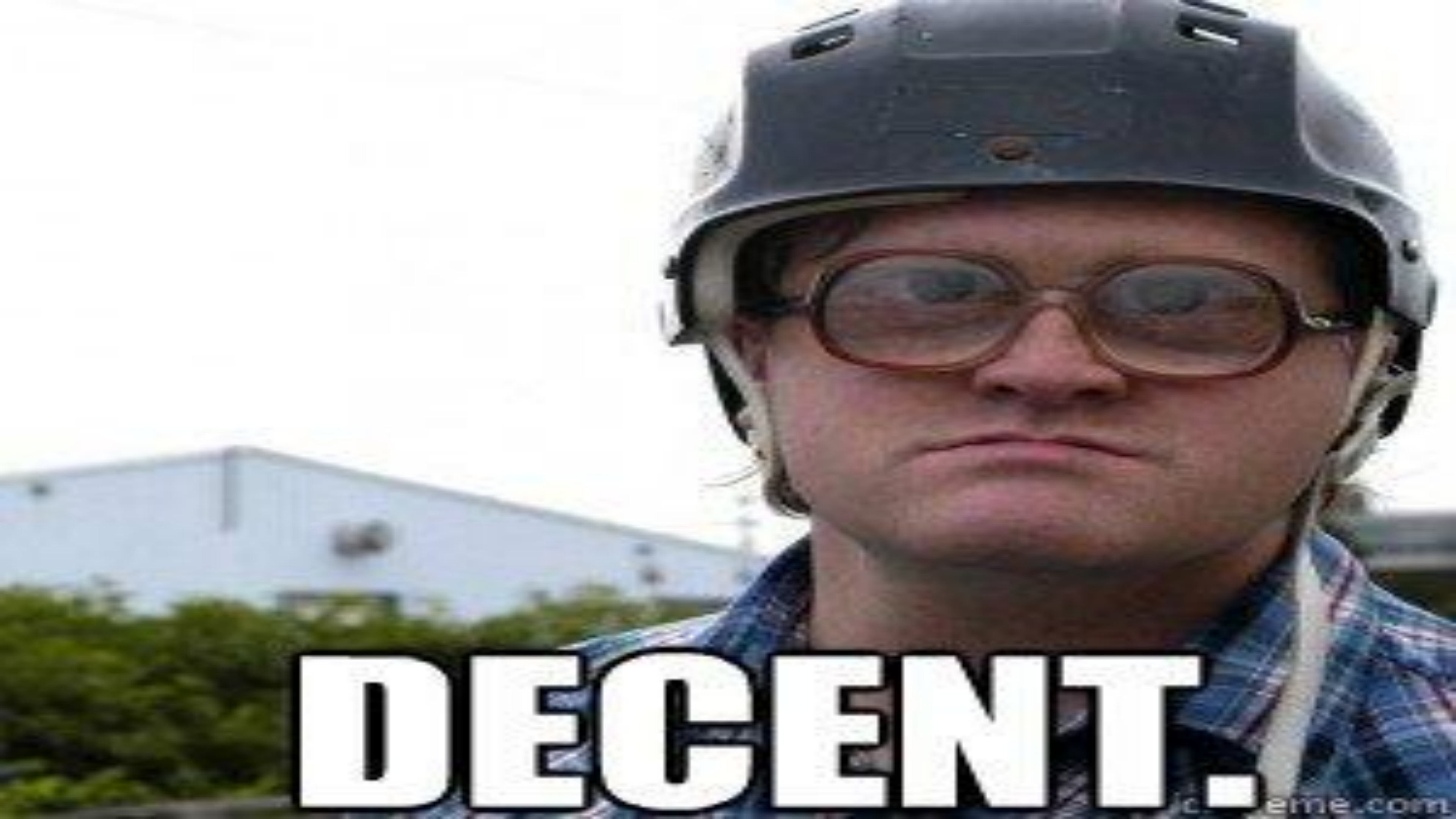
How do they work?

let’s use a Record Tab PI as our example

- Fetch all PI on page load
- Block rendering of Record tabs until PI are ran
- Pass an array of objects that represent the tabs to render to PI
- Execute PI in order, asynchronously
- PI modify the objects inside the array
- Record tabs are displayed

How are they managed?


Bullhorn
Find
Add
Privacy
Help
Johnathan Novo III
JN
Menu
Parse Resume
Close All Tabs


DECENT.

Novo-Elements

ENGAGE

What is new?

- Data Tables
- Accordion
- Enhanced Pickers / Forms
- A lot more!!

Change Pagination Style

Standard | Basic

Toggle Global Search

Show | Hide

Configure Columns

CONFIGURE COLUMNS

Configure Columns

SHOW ROW DETAILS (FIRST TABLE) | HIDE ROW DETAILS (FIRST TABLE)

PASSING AN ARRAY OF ROWS

ITEMS PER PAGE: 10 — 1 2 3 4 5 — Total: 1000 — REFRESH — ACTIONS

			ID	Status	Enabled
		ACTIONS	0	(1) Status 0	true
		ACTIONS	1	(1) Status 1	false
		ACTIONS	2	(1) Status 2	true
		ACTIONS	3	(1) Status 3	false
		ACTIONS	4	(1) Status 4	true
		ACTIONS	5	(1) Status 5	false
		ACTIONS	6	(1) Status 6	true
		ACTIONS	7	(1) Status 7	false
		ACTIONS	8	(1) Status 8	true
		ACTIONS	9	(1) Status 9	false

THEMED EXPANSION PANEL

This is an example of a themed list.

Contacts

Job Orders

```
<novo-expansion-panel theme="contact" padding="false">
    <novo-expansion-panel-header>
        <novo-panel-title>
            <novo-icon>person</novo-icon>
            <label>Contacts</label>
        </novo-panel-title>
    </novo-expansion-panel-header>

    <novo-table class="table-condensed"
        [rows]="details.rows"
        [columns]="details.columns"
        [config]="details.config"
        rowIdentifier="name"></novo-table>
```

Lots more to come!

- Angular 6!
- Revamped demos / examples!
- Much more!!!

http://bullhorn.github.io/novo-elements/

Bullhorn CLI

ENGAGE

Why did we build it?

bullhorn [namespace] <action>

- Accelerate Development
- Standardize Workflow
- Build System Support (Jenkins)
- **Extension Management**

What can it do?

bullhorn [namespace] <action>

- Accelerate Development
- Standardize Workflow
- Build System Support (Jenkins)
- **Extension Management**

HOW DOES it work?

Installation

```
➜ npm install -g @bullhorn/bullhorn-cli
```

```
➜ bullhorn --version

➜ bullhorn --help
```

Authentication

```
➜ bullhorn auth login
? Username:  bkimball
? Password:  [hidden]
Authorizing...
Authorization Complete: bf0fce1b-cdbb-4503-be49-700921b58961


➜ bullhorn auth login -u bkimball -p [hidden]
```

Configuration

```
➜ bullhorn config get environment
environment was set to https://universal.bullhornstaffing.com
```

```
➜ bullhorn config set environment https://universal.bullhorns…
environment was set to https://universal.bullhornstaffing.com
```

Extensions

```
➜ bullhorn extensions --help
  Usage: extensions [options] [command] <action>
    commands to manage extensions

  Options:
    -h, --help   output usage information

  Commands:
    extract      Extract an extension from the extension config JSON file
    list         list installed extensions
    upload       Upload an extension after extracting
```

Extensions List

```
➜ bullhorn extensions list
finding extensions...

id  name                       description                   numExts
--  -------------------------  ----------------------------  -------
50  Shift Scheduling           null                          3
55  Test Extension             null                          1
56  Integration Test           null                          2
67  PS-Test 1                  null                          5
74  Custom Cards               null                          2
```

Typings for TypeScript Support

➜ `bullhorn` typings generate

Or

➜ `npm` install --save @bullhorn/bullhorn-types


"That's all Folks!"

➜ `bullhorn` extensions *

➜ `bullhorn` functions *

➜ `bullhorn` events *

➜ `bullhorn` data *

https://github.com/bullhorn/bullhorn-cli

ENGAGE

Questions?


MAKE YOUR NIGHTS
16°C
AR BOA
SÓ ÔNIBUS
6 - 22h
Mantenha
Sua Cidade Limpa
GAP