

---

# **Survos Documentation**

***Release 6.0***

**Tac Tacelosky & Keith Ivey**

**Jan 24, 2020**



---

## Contents

---

<b>1</b>	<b>Getting Started with Modules</b>	<b>3</b>
<b>2</b>	<b>Recruit</b>	<b>5</b>
<b>3</b>	<b>Study</b>	<b>7</b>
<b>4</b>	<b>Indexes and tables</b>	<b>75</b>



The Survos Platform provides researchers with powerful tools to interact with their study participants. Participants can interact via random or fixed-interval SMS surveys, provide detailed, continuous information about their location, and collect photos via SMS or our app. Researchers can extract information about those photos via Amazon's Mechanical Turk, a crowd-sourcing platform.

This documentation provides detailed instructions about what the tools do, and how they are configured. As with many comprehensive systems, sometimes getting started is the hardest part. The major parts of the data collection system are broken down into "Modules". The basic workflow is the same for all the survey types.



---

## Getting Started with Modules

---

The platform has 4 related but independent *modules*. Each module collects a different type of data. All modules consist of protocols, surveys, waves, tasks, assignments, sets and channels. These terms will be defined within each module, with examples. Knowing these terms and how they are connected to each module is key to understanding the platform.

What can you do with the Survos Platform? These four verbs answer that question, and are the names of the modules:

- **Recruit** recruit and enroll study participants or field workers
- **Study** collect personal data about a set of study participants
- **Audit** get objective (not personal) data about a set of inanimate objects, like images or places
- **Annotate** combination of Study and Audit, collects *personal data* about a set of places or events, basically a personal map or calendar annotation.

In general, the workflow to collect data within any module is as follows:

1. Establish a *set* of who or what those questions will be asked of
2. Configure a *protocol* associated with that set of data and a channel type for how that data will be collected.
3. Design one or more *surveys* – the questions you want to ask
4. Add one or more surveys and timing information to the protocol, this is called a deployment wave, or simply *wave*.
5. Once the protocol has been configured with surveys and timing information, activate that protocol.





## CHAPTER 2

---

### Recruit

---

---

**Important:** Recruit and enroll study participants. The purpose of this module is to create new participant accounts within the system.

---

**Protocol** The protocol configuration includes how the candidate will take the survey. In most cases, this will be either self-enrolled via the participant website, or done as an interview by a researcher who will enter the data via the administrative website.

**Survey** A list of questions and answer options. Answers can have scores associated with them, when a candidate fills out a survey, answers with scores are totaled and can be used in the wave configuration to automatically approve those candidates as participants.

**Wave** The deployment of the survey questions, along with the configuration data specific to automatic enrollment, like the minimum score required.

Most of the example projects use a self-enrollment protocol, with very minimal requirements for actual enrollment.

A typical recruit protocol might be to ask if someone smokes cigarettes, is above 18, and has not tried to quit in the last year. Based on their answers, they may be accepted in the study. At that point, they'd have an account tagged as active, and therefore would be added into the "accepted participants" set (used by Study protocols).



---

**Tip:** Collect personal data about a set of study participants. Study participants are created in the Recruit module, or by simply creating a participant account.

---

**Participant Set** The collection of participants who will be enrolled in each wave of this protocol. The default set is “All Accepted Participants”, or you can create a subset like “All participants who self-identify as smokers” or “All participants who have completed with baseline protocols” or “All participants who have completed the observation period and have a compliance score of greater than 80%”.

**Protocol** The protocol configuration includes the participant set, the schedule type (e.g. just one time, like a baseline survey, or periodic, like a daily diary, or even a testing schedule, called a “blitz”). If the schedule type is periodic, then the duration of the protocol is also defined. The method of communication is also defined in the protocol, e.g. SMS or web-based. For example, you can configure the “Observation Protocol” as a 2-week period communicating with pregnant smokers via SMS. Note that when you initially set up the protocol, there are no surveys or schedules yet – those come when the wave is configured.

**Survey** A list of questions and answer options. For protocols with schedule type of for one-time surveys (e.g. a baseline protocol), these questions are likely to be in the form of “Have you ever...?” or “Do you...?” For periodic protocols, like an observation period, the questions are likely be EMA (Ecological Momentary Assessments, “in-the-moment” surveys), or coverage surveys, like a daily diary. An EMA survey may have questions in the form “Right now, are you...?”, e.g. “Right now, are you with other people?”, “Right now, how much do you want to use marijuana?”. A coverage survey typically contains questions like “Thinking about today only, did you purchase any tobacco products?” or “In the last 4 hours, did you exercise?” The common theme of surveys in the Study Module is that they are questions about the participant.

**Wave** The deployment of the survey questions within a protocol, along with relevant configuration information. If it’s part of a periodic protocol, then the specific schedule is defined here. Also, the communication channel is defined and the prompt channel (email, SMS, etc.). Compliance thresholds are set, expiration times, reminders, etc. The wave is where the majority of the settings are configured, and are defined in detail in this documentation.

A typical study protocol might be for 3 week to send SMS prompts 4 times per day asking participants who have completed the baseline protocol if they have smoked in the last 4 hours, as well as random prompts 6 times per day to ask if they are smoking right now, and if not, what their craving to smoke is.

Contents:

Here are some terms that are common to the Survos Platform.

## 3.1 Terminology (short and sweet)

Survey/Survey Instrument: *what* you ask Protocol: *time period* and *to whom* you deploy Survey Instruments Wave: *how* you deploy Survey Instruments Account/Member: A participant (or other role) in a study Task: The instance of a wave associated with specific data. Assignment: The participant's survey answers associated with a Task

## 3.2 Terminology (detailed)

**Project/Study** The data, people and protocols of a research study.

**Survey** A list of questions and answer options, tied to a how those questions will be deployed, such as via SMS or Turk or a Web Interview

**Module** A collection of protocols that operate on the same *source data*. The primary modules are **Study** (collects data about study participants), **Audit** (collects objective, no personal, data about things, like images or places, **Annotations** (combination of Study and Audit, collects *personal data* about a *place*, basically a personal map annotation.

**Protocol** A collection of survey types that collect data in a similar way. Because the types of data the system can collect is so varied, selectively installing modules allows the system to be navigated more easily. Many projects only use one or two [ref:modules](#).

**Wave** Deployment of the survey questions. It is the central object in the system, and is the configuration for who will answer the survey questions, when, how, how often, how much they'll be paid, etc. There are many types of jobs, and each type will have its own specific configuration, which will be described later in the document.

**Protocol** A group of waves that share a common duration and criteria.

**Turk** Amazon Mechanical Turk (AMT) is a marketplace for crowd-sourced work. The Posse Platform enables a user to create what AMT calls external HITs, which are tasks run outside of the AMT system. Internal hits for AMT are somewhat limited. With a Turk job, a task is posted and a crowd of anonymous users has a chance to accept individual assignments and complete the task. This is particularly good for gathering data about a photo, where if a group of anonymous people reach consensus about something, it is likely to be valid.

- **Accounts (aka “member”):** Accounts associated with a project. All users have an identifier that is unique with in the project, for study participants, this is sometimes referred to as a “Subject ID”, and except when testing, this identifier should not be personally identifiable. There are various roles users have, such as “applicant”, “participant”, “turker” “researcher” and “participant”.
- **Assignment/Survey Response:** The participant's survey answers.
- **Data:** Each project may contain its own database of places, images and other data. Jobs that use data (e.g. visiting a list of places, looking at a list of photos) get their data from these tables. Wave collect data (e.g. taking pictures) put their results here (so the results can be used by subsequent Surveys).
- **Import:** Import data from external source (e.g. csv file, CartoDB)

## 3.3 Data Collection Workflow

All data collection within a project works roughly the same way, and will be described in specific detail in each module.

1. Within a module, decide *how* your questions will be delivered (e.g. SMS, Interview, Turk), and create a survey.
2. Design your questions within the survey, paying special attention to the question codes, types and possible responses. The data collected will be available using these codes.
3. Define the specifics of your survey deployment, called a `_wave_` – this will change for each specific survey type, but is basically *who* receives the survey, *when* and how often.
4. Deploy the wave and collect the data. Wave Configuration is specific to each survey type, and is described in detail under “Deployment” for each Survey Type

## 3.4 Lifecycles / Workflows

Each lifecycle component shares a `state_machine`.

Contents:

### 3.4.1 Workflow project in Project

#### Project States

- configuration
- pending\_approval
- collecting\_data
- aborted
- complete
- cancelled

#### Project Transitions

1. propose
2. reject
3. cancel
4. launch
5. stop\_at\_deadline
6. abort
7. all\_data\_collected

#### The Workflow

### 3.4.2 Workflow audit\_task in Task

#### Task States

- created

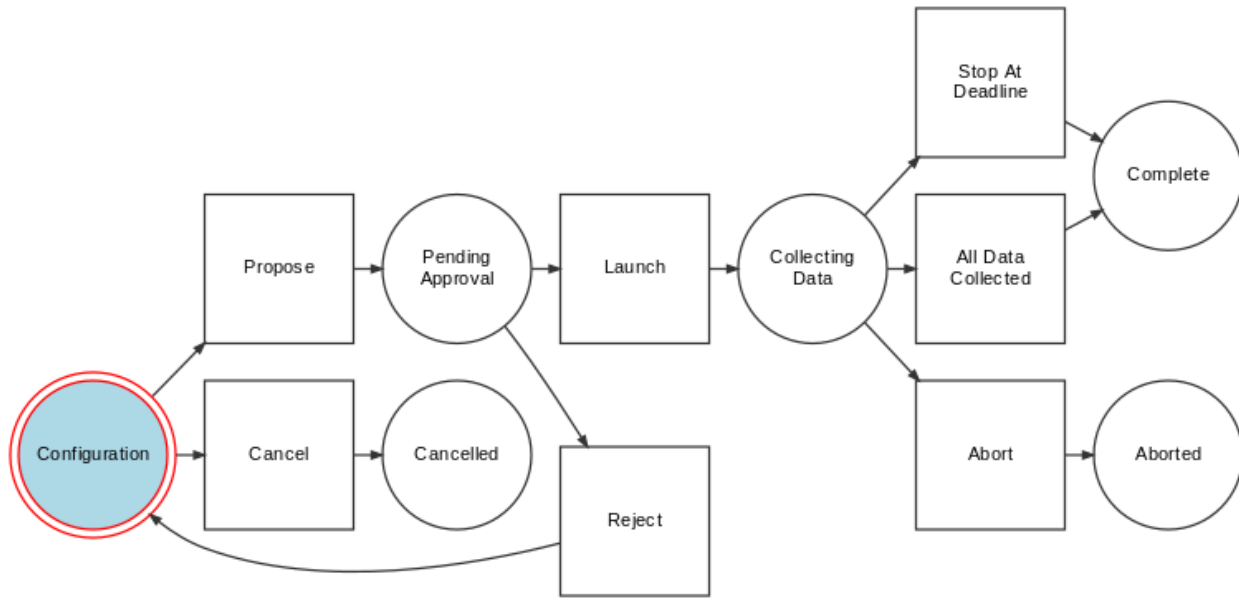


Fig. 1: Workflow project

- published
- publish\_error
- expired
- complete
- Assignable

## Task Transitions

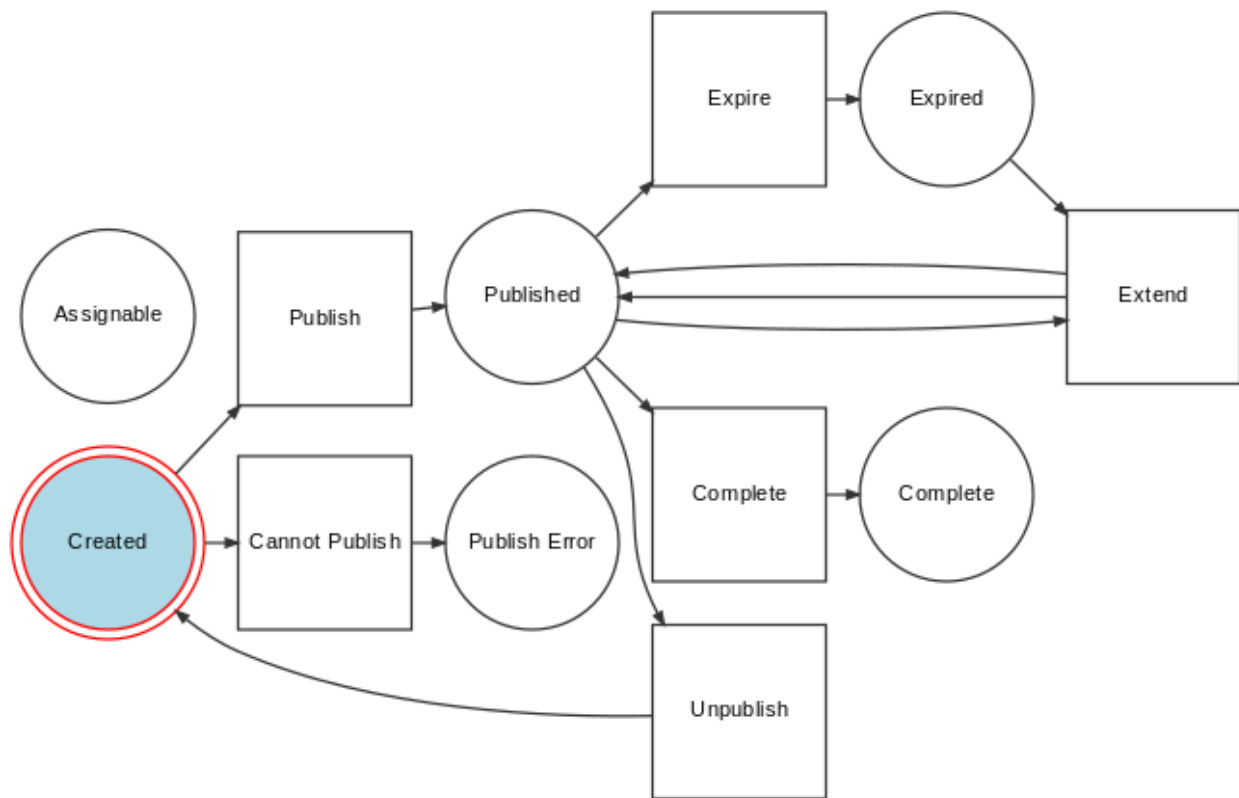
1. publish
2. unpublish
3. cannot\_publish
4. expire
5. extend
6. complete

## The Workflow

### 3.4.3 Workflow task\_enrollment in Task

#### Task States

- created
- enrolled
- dropout



- dismissed
- error
- finished
- deleted

### Task Transitions

1. enroll
2. admin\_dismiss
3. last\_day
4. participant\_quits
5. admin\_delete
6. error
7. fix\_error

### The Workflow

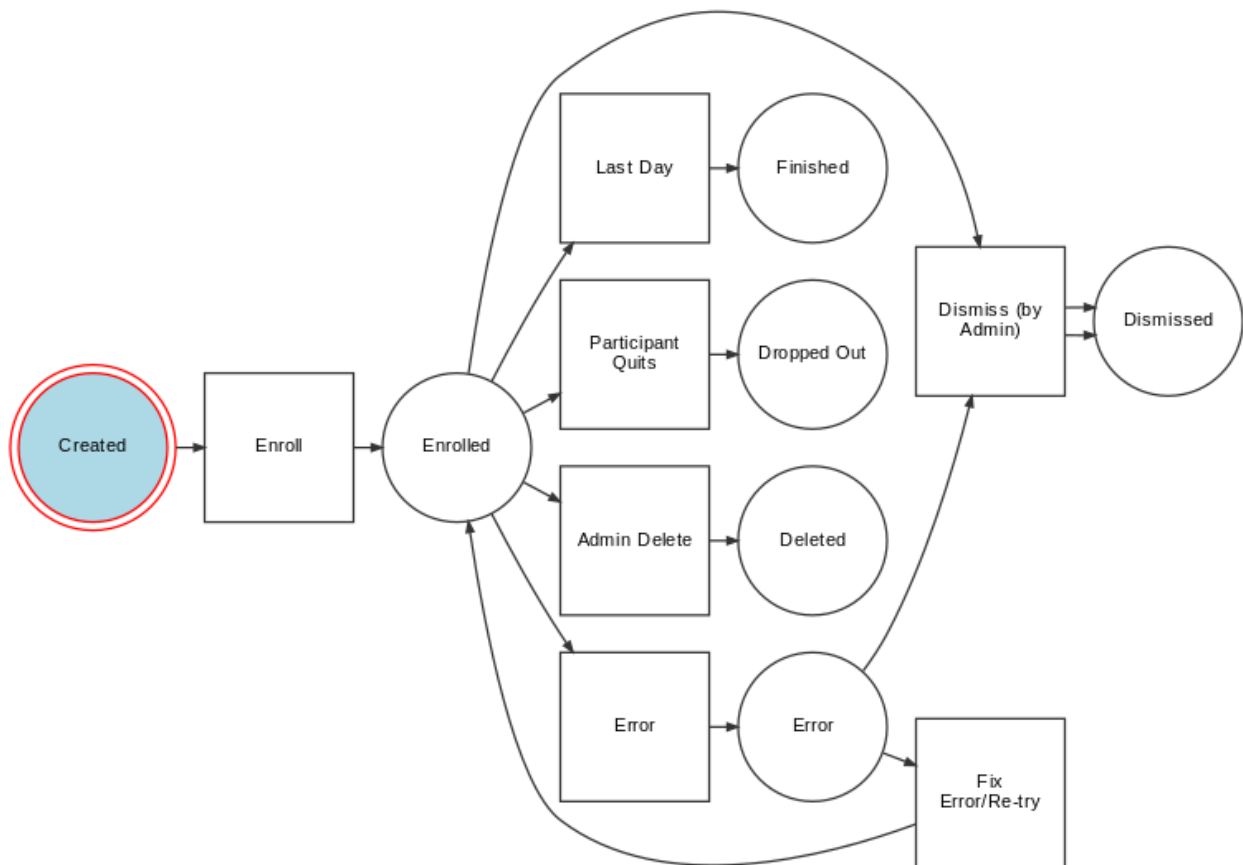


Fig. 3: Workflow task\_enrollment



### 3.4.4 Workflow outgoing\_msg in ChannelMessage

#### ChannelMessage States

- created
- dispatched
- delivered
- error

#### ChannelMessage Transitions

1. send\_to\_dispatcher
2. dispatch\_error
3. deliver

#### The Workflow

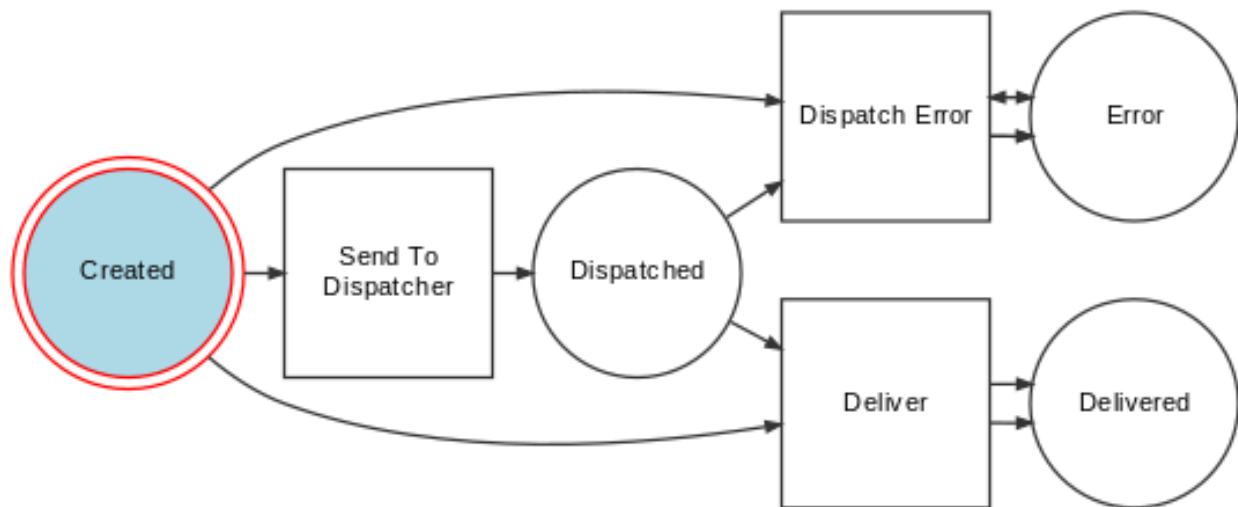


Fig. 4: Workflow outgoing\_msg

### 3.4.5 Workflow survey\_response in Assignment

#### Assignment States

- scheduled
- initiated
- responding
- abandoned
- complete

- expired

## Assignment Transitions

1. at\_start\_time
2. single\_submission
3. reminders\_sent
4. warnings\_sent
5. never\_responds
6. first\_response
7. stops\_responding
8. no\_more\_questions

## The Workflow

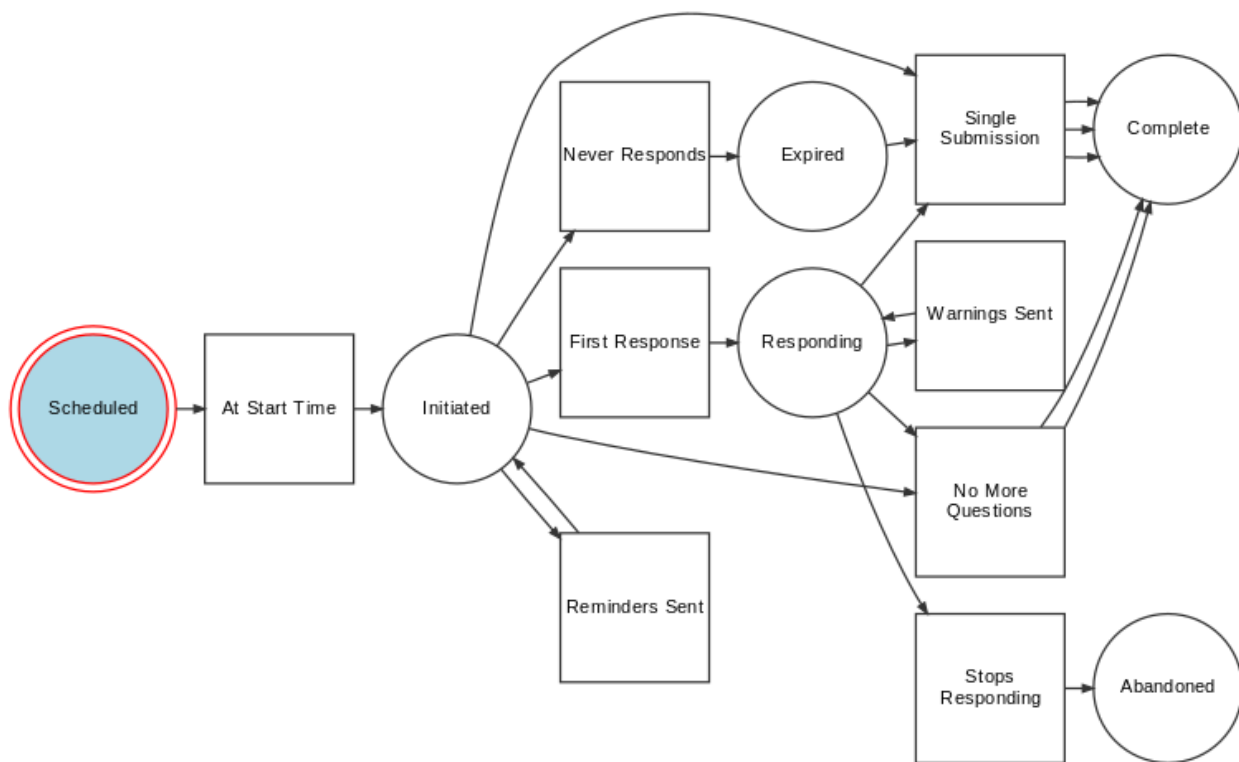


Fig. 5: Workflow survey\_response

## 3.4.6 Workflow member\_participant in Member

### Member States

- visitor

- applicant
- ineligible
- eligible
- active
- finished
- dismissed

## Member Transitions

1. applicant\_fills\_out\_screener
2. admin\_creates\_participant
3. accept\_eligibility
4. deny\_eligibility
5. approve\_application
6. enroll\_in\_waves
7. finish
8. dismiss

## The Workflow

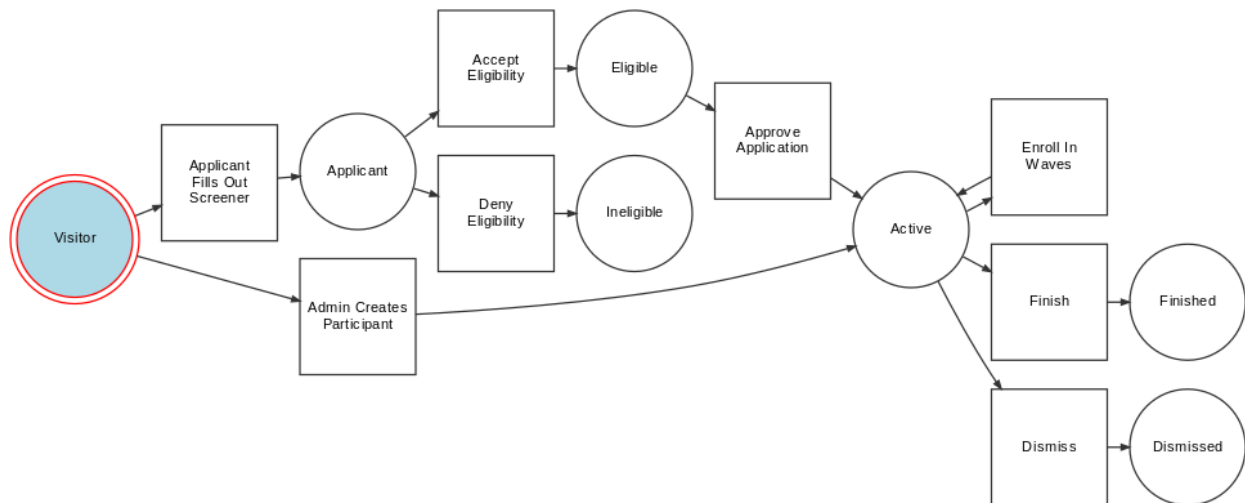


Fig. 6: Workflow member\_participant

### 3.4.7 Workflow data in Data

#### Data States

- record\_created

- pending
- ready
- error

### Data Transitions

1. request\_processing
2. processing\_success
3. processing\_failed

### The Workflow

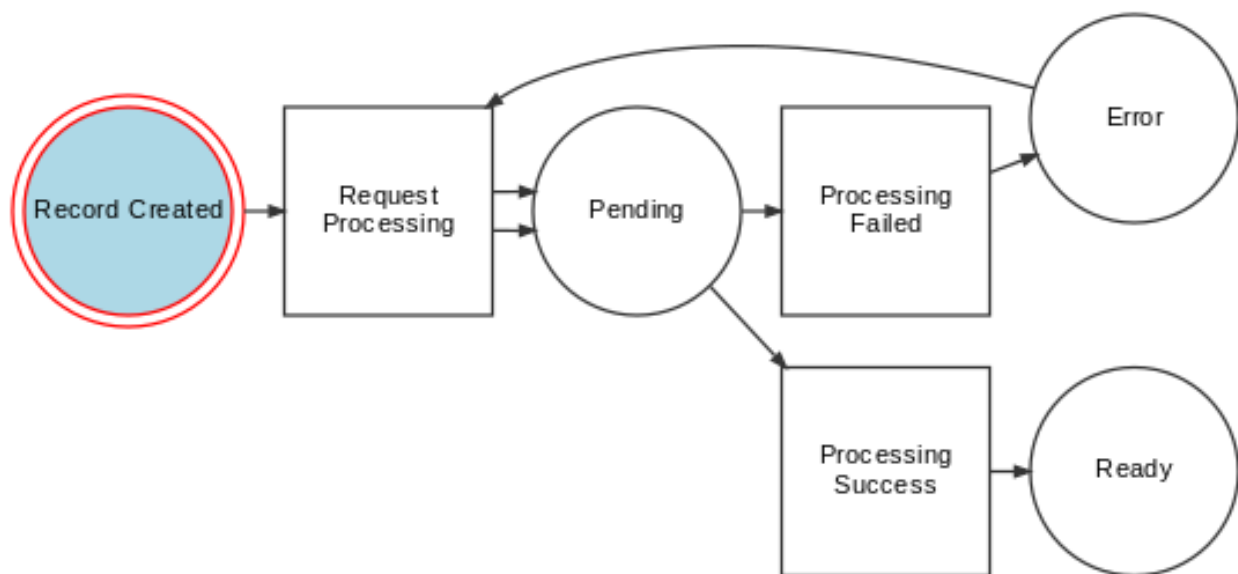


Fig. 7: Workflow data

## 3.4.8 Workflow survey in Survey

### Survey States

- new
- configured
- draft
- survey\_preview
- waves\_active
- data\_collected
- no\_data\_collected
- locked

- paused
- archived

## Survey Transitions

1. configure
2. design\_questions
3. modify\_configuration
4. publish
5. deploy\_waves
6. data\_collected
7. unlock
8. edit
9. archive
10. pause
11. resume

## The Workflow

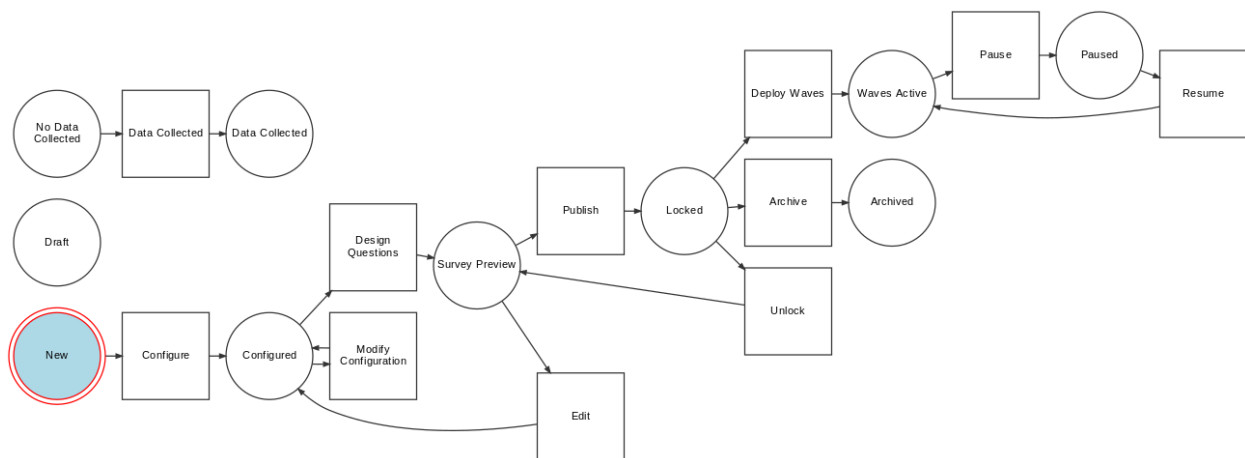


Fig. 8: Workflow survey

### 3.4.9 Workflow protocol in Protocol

#### Protocol States

- configuration
- active
- paused

- stopped

## Protocol Transitions

1. activate
2. reactivate
3. restart
4. pause
5. stop
6. reset

## The Workflow

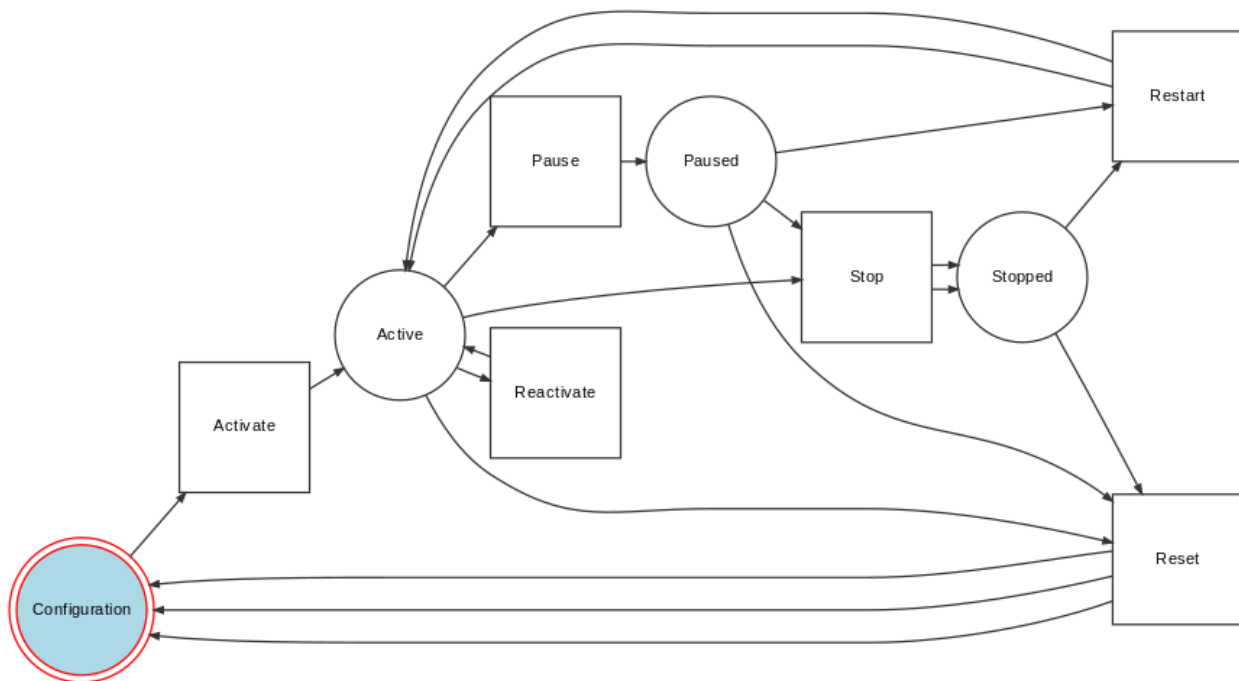


Fig. 9: Workflow protocol

### 3.4.10 Workflow export\_file in ExportFile

#### ExportFile States

- queued
- processing
- complete
- ignored
- failure

## ExportFile Transitions

1. activate
2. ignore
3. finish
4. error
5. restart

## The Workflow

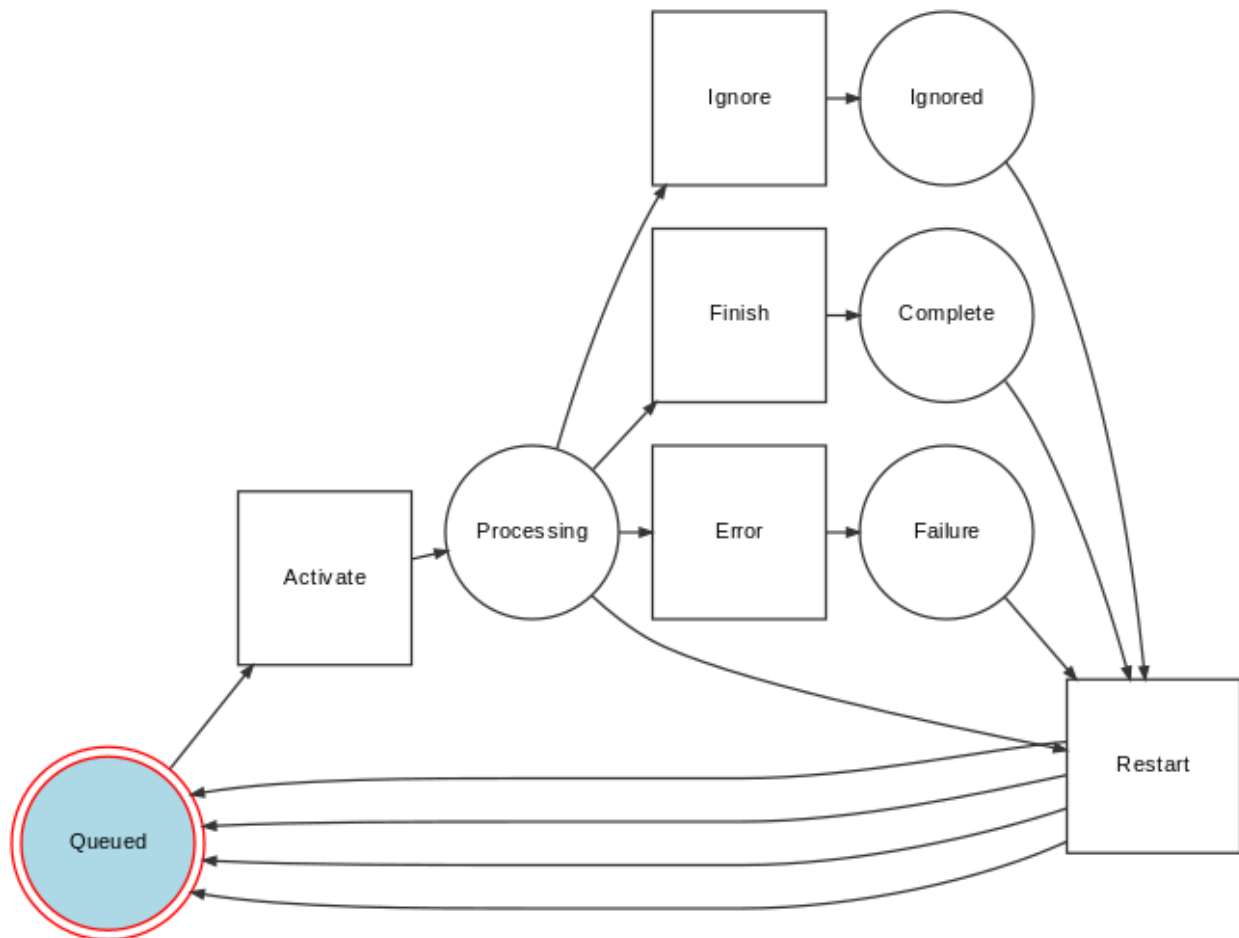


Fig. 10: Workflow export\_file

## 3.5 Account Roles

### 3.5.1 *ROLE\_PARTICIPANT*

Participants are study subjects in a study. They have a unique anonymous identifier.

### 3.5.2 *ROLE\_OWNER*

Owner accounts have full access to configure and edit a study. They also have full access to PII. Their account identifier (username) is identifiable, even if they are participating as a study participant (usually for testing)

### 3.5.3 *ROLE\_RESEARCHER*

Researchers have read-only access to the entire study design and data, EXCEPT PII of the participants. Their account identifier (username) is identifiable, even if they are participating as a study participant (usually for testing)

### 3.5.4 *ROLE\_APPLICANT*

Applicants are not yet in the study, and have no study identifier. They are not eligible to be enrolled in any waves except screener waves, which are used to determine if they are eligible for the study.

### 3.5.5 *ROLE\_FIELD\_WORKER*

roles.ROLE\_FIELD\_WORKER

## 3.6 Modules

Contents:

### 3.6.1 Recruit Survey Types

Information to screen applicants (potential study participants or field workers)

#### What's Needed to Deploy

- Qualified Survey Participants
- Images

#### Use Case

#### Integration Types

#### Typical Questions

#### Survey Types/Ways to Administer

- WebInterview, see *WebInterview Integration Types*
- ObserveWeb, see *ObserveWeb Integration Types*



## Survey Participants

Administrators can invite or add participants. Projects with “Allow Self Registration” and an active Screener Wave allow visitors to create an fill out a pre-qualification survey.

## Data Collected

## Technical Notes

Code: screener

## WebInterview Integration Types

WebInterview

The purpose of this integration is to provide a channel to collect data.

## WebInterview Wave Configuration

Go through each screen of the Wave Form or Wizard for a WebInterview Wave

## WebInterview Basic

This is the form snippet from [basic Tab in the WebInterview Wave](#).

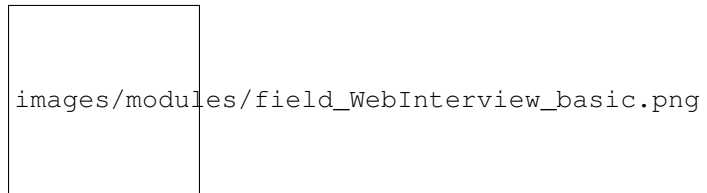


Fig. 11: Rendered Form WebInterview Wave Tab basic

## WebInterview Questions

This is the form snippet from [questions Tab in the WebInterview Wave](#).

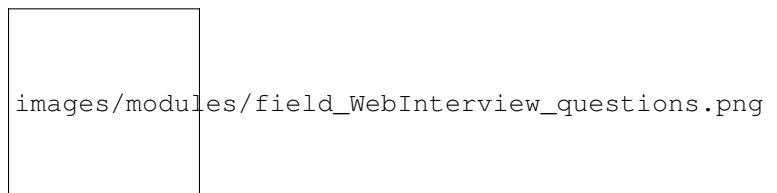


Fig. 12: Rendered Form WebInterview Wave Tab questions

### WebInterview Schedule

This is the form snippet from `schedule` Tab in the `WebInterview Wave`.

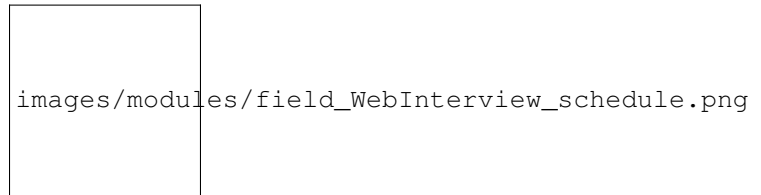


Fig. 13: Rendered Form WebInterview Wave Tab schedule

### WebInterview Enrollment

This is the form snippet from `enrollment` Tab in the `WebInterview Wave`.

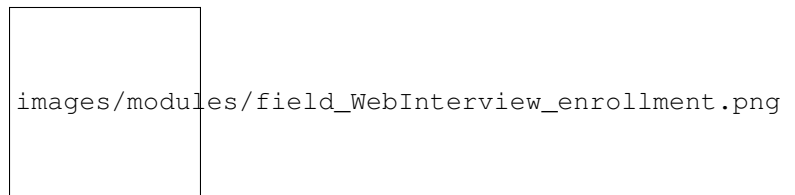


Fig. 14: Rendered Form WebInterview Wave Tab enrollment

### WebInterview Task

This is the form snippet from `task` Tab in the `WebInterview Wave`.

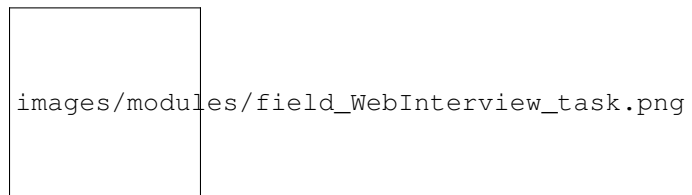


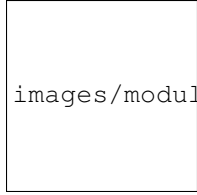
Fig. 15: Rendered Form WebInterview Wave Tab task

### WebInterview Expiration

This is the form snippet from `expiration` Tab in the `WebInterview Wave`.

### WebInterview Messages

This is the form snippet from `messages` Tab in the `WebInterview Wave`.



images/modules/field\_WebInterview\_expiration.png

Fig. 16: Rendered Form WebInterview Wave Tab expiration



images/modules/field\_WebInterview\_messages.png

Fig. 17: Rendered Form WebInterview Wave Tab messages

## ObserveWeb Integration Types

### ObserveWeb


The purpose of this integration is to provide a channel to collect data.

### ObserveWeb Wave Configuration

Go through each screen of the Wave Form or Wizard for a ObserveWeb Wave

### ObserveWeb Basic

This is the form snippet from [basic Tab in the ObserveWeb Wave](#).



images/modules/field\_ObserveWeb\_basic.png

Fig. 18: Rendered Form ObserveWeb Wave Tab basic

### ObserveWeb Questions

This is the form snippet from [questions Tab in the ObserveWeb Wave](#).

### ObserveWeb Schedule

This is the form snippet from [schedule Tab in the ObserveWeb Wave](#).

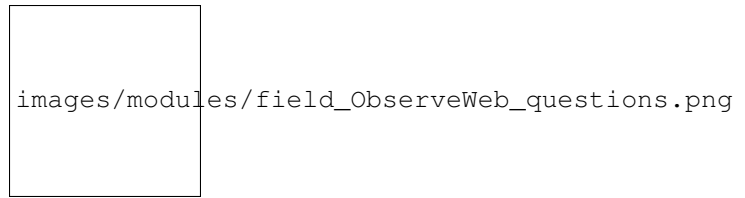


Fig. 19: Rendered Form ObserveWeb Wave Tab questions

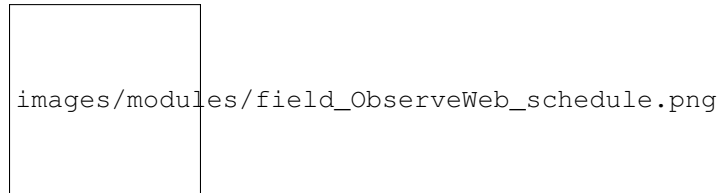


Fig. 20: Rendered Form ObserveWeb Wave Tab schedule

### ObserveWeb Enrollment

This is the form snippet from [enrollment Tab](#) in the ObserveWeb Wave.

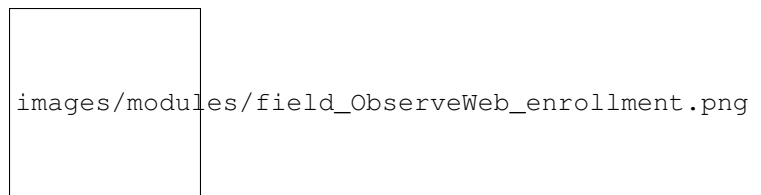


Fig. 21: Rendered Form ObserveWeb Wave Tab enrollment

### ObserveWeb Task

This is the form snippet from [task Tab](#) in the ObserveWeb Wave.

### ObserveWeb Expiration

This is the form snippet from [expiration Tab](#) in the ObserveWeb Wave.

### ObserveWeb Messages

This is the form snippet from [messages Tab](#) in the ObserveWeb Wave.

## 3.6.2 Study Survey Types

Personal data about registered study participants

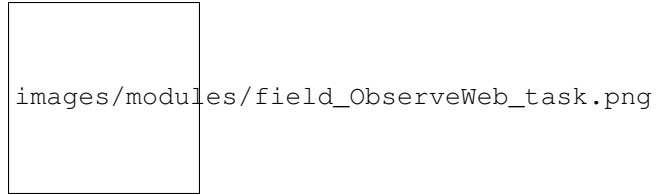


Fig. 22: Rendered Form ObserveWeb Wave Tab task

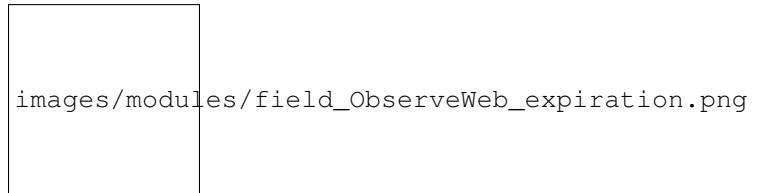


Fig. 23: Rendered Form ObserveWeb Wave Tab expiration

### What's Needed to Deploy

- Qualified Survey Participants
- Images

### Use Case

When you want to repeatedly collect data from a participant over a period of time. You can configure a daily diary – a scheduled once-a-day survey, or ask participants to self-report every time they engage in some behavior, or randomly prompt participants with mood or behavior questions.

### Integration Types

### Typical Questions

- How many hours did you sleep last night?
- How many cigarettes have you smoked in the last 2 hours?
- On a scale of 0 to 5, how much are you craving a cigarette right now?
- Take a picture of your store receipt.



Fig. 24: Rendered Form ObserveWeb Wave Tab messages

## Survey Types/Ways to Administer

- ObserveWeb, see *ObserveWeb Integration Types*
- WebInterview, see *WebInterview Integration Types*
- TwilioSms, see *TwilioSms Integration Types*
- Mattermost, see *Mattermost Integration Types*
- Email, see *Email Integration Types*
- Simulator, see *Simulator Integration Types*
- Api, see *Api Integration Types*
- JsonResponses, see *JsonResponses Integration Types*

## Survey Participants

Administrators can invite or add participants. Projects with “Allow Self Registration” and an active Screener Wave allow visitors to create an fill out a pre-qualification survey.

## Data Collected

## Technical Notes

Code: ongoing

## TwilioSms Integration Types

TwilioSms

The purpose of this integration is to provide a channel to collect data.

## TwilioSms Wave Configuration

Go through each screen of the Wave Form or Wizard for a TwilioSms Wave

## TwilioSms Basic

This is the form snippet from [basic Tab in the TwilioSms Wave](#).

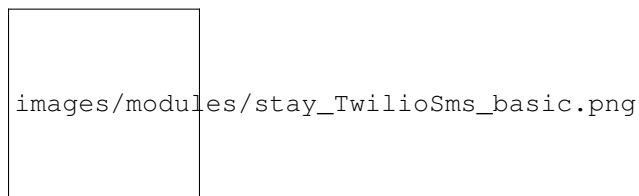


Fig. 25: Rendered Form TwilioSms Wave Tab basic

## TwilioSms Enrollment

This is the form snippet from `enrollment` Tab in the TwilioSms Wave.

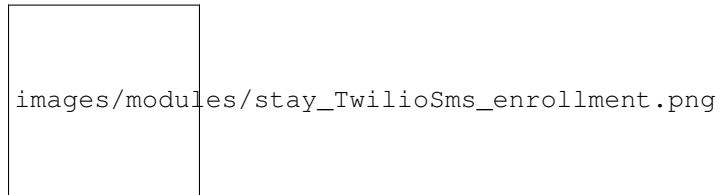


Fig. 26: Rendered Form TwilioSms Wave Tab enrollment

## TwilioSms Task

This is the form snippet from `task` Tab in the TwilioSms Wave.

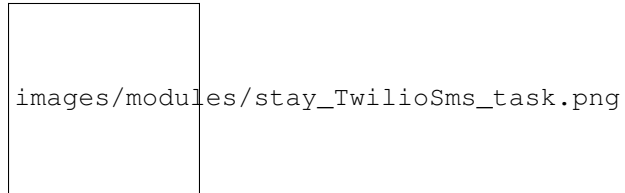


Fig. 27: Rendered Form TwilioSms Wave Tab task

## TwilioSms Expiration

This is the form snippet from `expiration` Tab in the TwilioSms Wave.

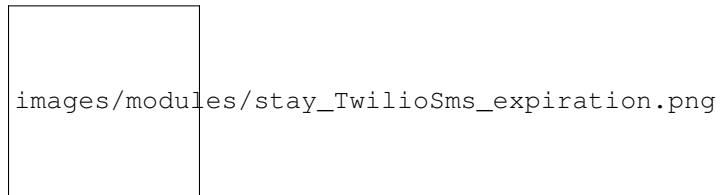


Fig. 28: Rendered Form TwilioSms Wave Tab expiration

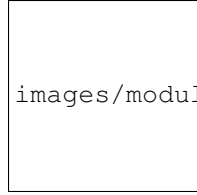
## TwilioSms Messages

This is the form snippet from `messages` Tab in the TwilioSms Wave.

## Mattermost Integration Types

Mattermost

The purpose of this integration is to provide a channel to collect data.



images/modules/stay\_TwilioSms\_messages.png

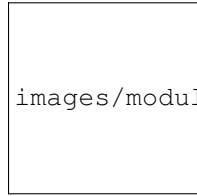
Fig. 29: Rendered Form TwilioSms Wave Tab messages

## Mattermost Wave Configuration

Go through each screen of the Wave Form or Wizard for a Mattermost Wave

### Mattermost Basic

This is the form snippet from [basic Tab in the Mattermost Wave](#).

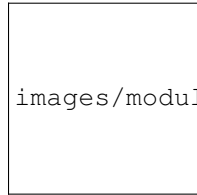


images/modules/stay\_Mattermost\_basic.png

Fig. 30: Rendered Form Mattermost Wave Tab basic

### Mattermost Enrollment

This is the form snippet from [enrollment Tab in the Mattermost Wave](#).



images/modules/stay\_Mattermost\_enrollment.png

Fig. 31: Rendered Form Mattermost Wave Tab enrollment

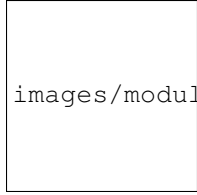
### Mattermost Task

This is the form snippet from [task Tab in the Mattermost Wave](#).

### Mattermost Expiration

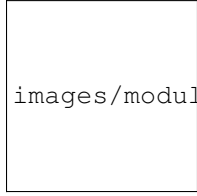
This is the form snippet from [expiration Tab in the Mattermost Wave](#).





images/modules/stay\_Mattermost\_task.png

Fig. 32: Rendered Form Mattermost Wave Tab task

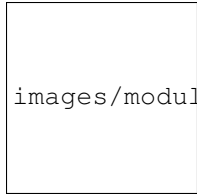


images/modules/stay\_Mattermost\_expiration.png

Fig. 33: Rendered Form Mattermost Wave Tab expiration

## Mattermost Messages

This is the form snippet from [messages Tab in the Mattermost Wave](#).



images/modules/stay\_Mattermost\_messages.png

Fig. 34: Rendered Form Mattermost Wave Tab messages

## Email Integration Types

Email

The purpose of this integration is to provide a channel to collect data.

## Email Wave Configuration

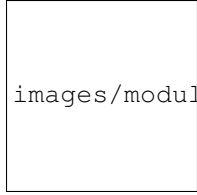
Go through each screen of the Wave Form or Wizard for a Email Wave

## Email Basic

This is the form snippet from [basic Tab in the Email Wave](#).

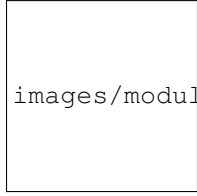
## Email Enrollment

This is the form snippet from [enrollment Tab in the Email Wave](#).



images/modules/stay\_Email\_basic.png

Fig. 35: Rendered Form Email Wave Tab basic



images/modules/stay\_Email\_enrollment.png

Fig. 36: Rendered Form Email Wave Tab enrollment

## Email Task

This is the form snippet from [task Tab in the Email Wave](#).



images/modules/stay\_Email\_task.png

Fig. 37: Rendered Form Email Wave Tab task

## Email Expiration

This is the form snippet from [expiration Tab in the Email Wave](#).

## Email Messages

This is the form snippet from [messages Tab in the Email Wave](#).

## Simulator Integration Types

Simulator

The purpose of this integration is to provide a channel to collect data.

## Simulator Wave Configuration

Go through each screen of the Wave Form or Wizard for a Simulator Wave

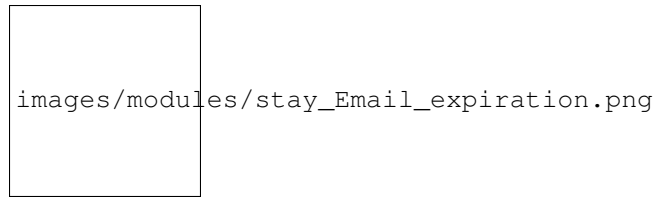


Fig. 38: Rendered Form Email Wave Tab expiration

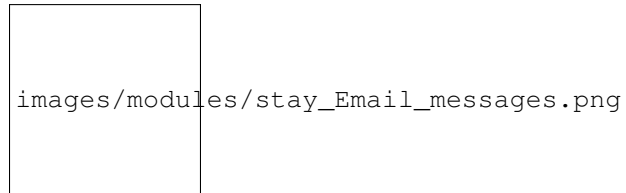


Fig. 39: Rendered Form Email Wave Tab messages

### Simulator Basic

This is the form snippet from [basic Tab in the Simulator Wave](#).

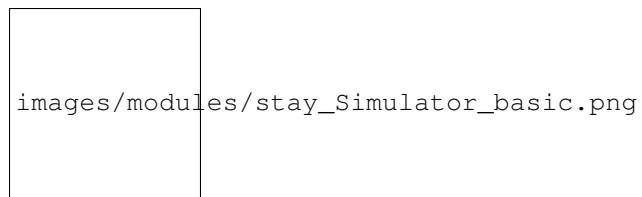


Fig. 40: Rendered Form Simulator Wave Tab basic

### Simulator Enrollment

This is the form snippet from [enrollment Tab in the Simulator Wave](#).

### Simulator Task

This is the form snippet from [task Tab in the Simulator Wave](#).

### Simulator Expiration

This is the form snippet from [expiration Tab in the Simulator Wave](#).

### Simulator Messages

This is the form snippet from [messages Tab in the Simulator Wave](#).

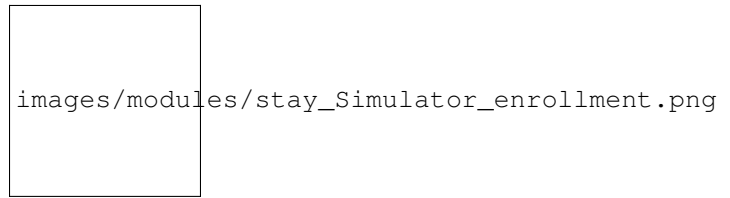


Fig. 41: Rendered Form Simulator Wave Tab enrollment

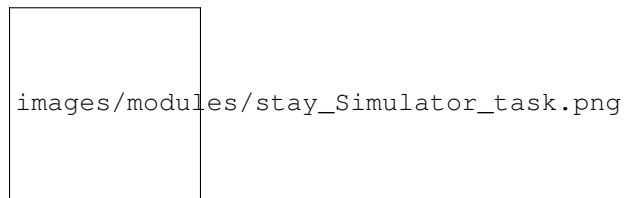


Fig. 42: Rendered Form Simulator Wave Tab task

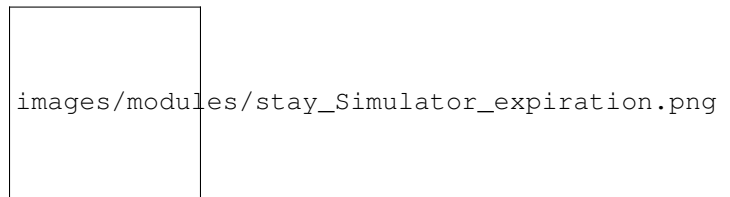


Fig. 43: Rendered Form Simulator Wave Tab expiration

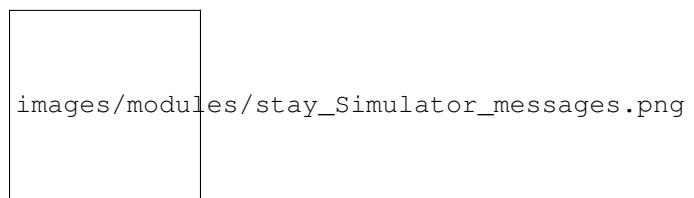


Fig. 44: Rendered Form Simulator Wave Tab messages

## Api Integration Types

### Api

The purpose of this integration is to provide a channel to collect data.

## Api Wave Configuration

Go through each screen of the Wave Form or Wizard for a Api Wave

### Api Basic

This is the form snippet from [basic Tab in the Api Wave](#).

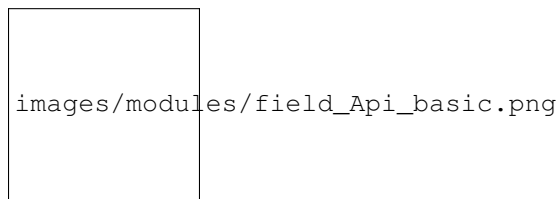


Fig. 45: Rendered Form Api Wave Tab basic

### Api Questions

This is the form snippet from [questions Tab in the Api Wave](#).

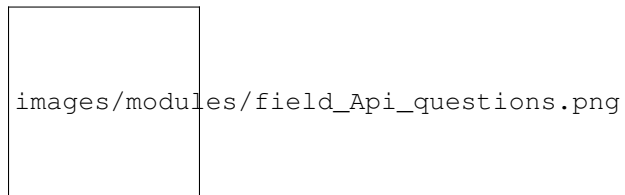


Fig. 46: Rendered Form Api Wave Tab questions

### Api Schedule

This is the form snippet from [schedule Tab in the Api Wave](#).

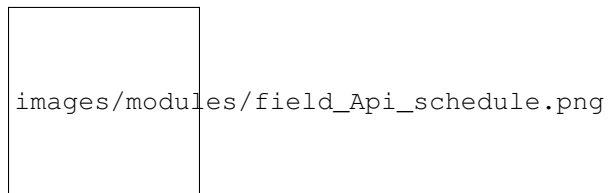


Fig. 47: Rendered Form Api Wave Tab schedule

## Api Enrollment

This is the form snippet from `enrollment` Tab in the `Api Wave`.

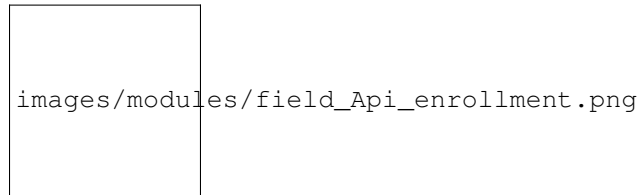


Fig. 48: Rendered Form Api Wave Tab enrollment

## Api Task

This is the form snippet from `task` Tab in the `Api Wave`.

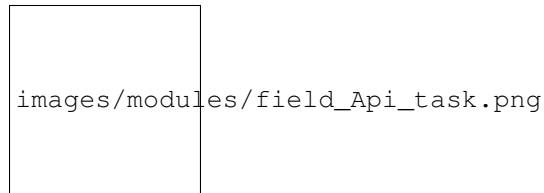


Fig. 49: Rendered Form Api Wave Tab task

## Api Expiration

This is the form snippet from `expiration` Tab in the `Api Wave`.

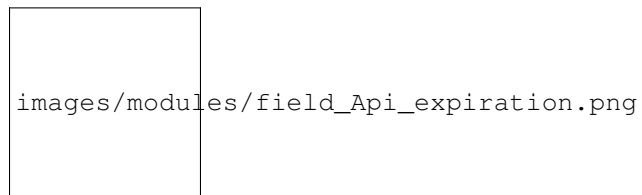


Fig. 50: Rendered Form Api Wave Tab expiration

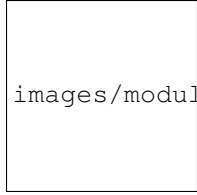
## Api Messages

This is the form snippet from `messages` Tab in the `Api Wave`.

## JsonResponses Integration Types

JsonResponses

The purpose of this integration is to provide a channel to collect data.



images/modules/field\_Api\_messages.png

Fig. 51: Rendered Form Api Wave Tab messages

## JsonResponses Wave Configuration

Go through each screen of the Wave Form or Wizard for a JsonResponses Wave

### JsonResponses Basic

This is the form snippet from [basic Tab in the JsonResponses Wave](#).

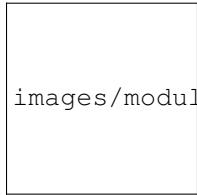


images/modules/field\_JsonResponses\_basic.png

Fig. 52: Rendered Form JsonResponses Wave Tab basic

### JsonResponses Questions

This is the form snippet from [questions Tab in the JsonResponses Wave](#).



images/modules/field\_JsonResponses\_questions.png

Fig. 53: Rendered Form JsonResponses Wave Tab questions

### JsonResponses Schedule

This is the form snippet from [schedule Tab in the JsonResponses Wave](#).

### JsonResponses Enrollment

This is the form snippet from [enrollment Tab in the JsonResponses Wave](#).



images/modules/field\_JsonResponses\_schedule.png

Fig. 54: Rendered Form JsonResponses Wave Tab schedule

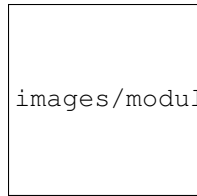


images/modules/field\_JsonResponses\_enrollment.png

Fig. 55: Rendered Form JsonResponses Wave Tab enrollment

### JsonResponses Task

This is the form snippet from `task` Tab in the `JsonResponses` Wave.



images/modules/field\_JsonResponses\_task.png

Fig. 56: Rendered Form JsonResponses Wave Tab task

### JsonResponses Expiration

This is the form snippet from `expiration` Tab in the `JsonResponses` Wave.

### JsonResponses Messages

This is the form snippet from `messages` Tab in the `JsonResponses` Wave.

## 3.6.3 Annotate Survey Types

Staypoints and events

### What's Needed to Deploy

- Qualified Survey Participants
- Images





Fig. 57: Rendered Form JsonResponses Wave Tab expiration



Fig. 58: Rendered Form JsonResponses Wave Tab messages

## Use Case

## Integration Types

## Typical Questions

## Survey Types/Ways to Administer

- Api, see *Api Integration Types*
- CartoDataSet, see *CartoDataSet Integration Types*
- CartoMap, see *CartoMap Integration Types*
- CollectedData, see *CollectedData Integration Types*
- Csv, see *Csv Integration Types*
- Email, see *Email Integration Types*
- Github, see *Github Integration Types*
- JsonResponses, see *JsonResponses Integration Types*
- Mapmob, see *Mapmob Integration Types*
- Mattermost, see *Mattermost Integration Types*
- ObserveWeb, see *ObserveWeb Integration Types*
- Participants, see *Participants Integration Types*
- PaypalRest, see *PaypalRest Integration Types*
- RabbitQueue, see *RabbitQueue Integration Types*
- RapidPro, see *RapidPro Integration Types*
- Simulator, see *Simulator Integration Types*
- SqsQueue, see *SqsQueue Integration Types*
- SubmittedAssignments, see *SubmittedAssignments Integration Types*

- Turk, see *Turk Integration Types*
- TwilioIvr, see *TwilioIvr Integration Types*
- TwilioSms, see *TwilioSms Integration Types*
- WebInterview, see *WebInterview Integration Types*

### Survey Participants

Administrators can invite or add participants. Projects with “Allow Self Registration” and an active Screener Wave allow visitors to create an fill out a pre-qualification survey.

### Data Collected

### Technical Notes

Code: stay

### CartoDataSet Integration Types

CartoDataSet

The purpose of this integration is to provide a channel to collect data.

### CartoDataSet Wave Configuration

Go through each screen of the Wave Form or Wizard for a CartoDataSet Wave

### CartoDataSet Basic

This is the form snippet from [basic Tab](#) in the [CartoDataSet Wave](#).

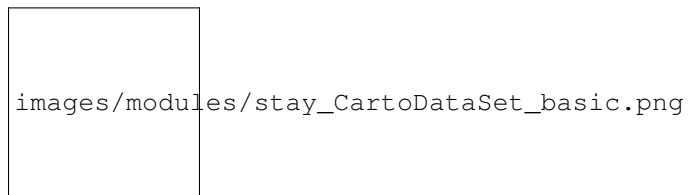


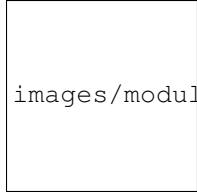
Fig. 59: Rendered Form CartoDataSet Wave Tab basic

### CartoDataSet Enrollment

This is the form snippet from [enrollment Tab](#) in the [CartoDataSet Wave](#).

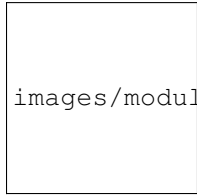
### CartoDataSet Task

This is the form snippet from [task Tab](#) in the [CartoDataSet Wave](#).



images/modules/stay\_CartoDataSet\_enrollment.png

Fig. 60: Rendered Form CartoDataSet Wave Tab enrollment

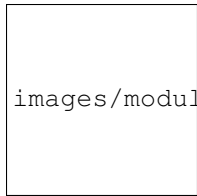


images/modules/stay\_CartoDataSet\_task.png

Fig. 61: Rendered Form CartoDataSet Wave Tab task

### CartoDataSet Expiration

This is the form snippet from [expiration Tab in the CartoDataSet Wave](#).



images/modules/stay\_CartoDataSet\_expiration.png

Fig. 62: Rendered Form CartoDataSet Wave Tab expiration

### CartoDataSet Messages

This is the form snippet from [messages Tab in the CartoDataSet Wave](#).

### CartoMap Integration Types

CartoMap

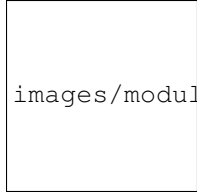
The purpose of this integration is to provide a channel to collect data.

### CartoMap Wave Configuration

Go through each screen of the Wave Form or Wizard for a CartoMap Wave

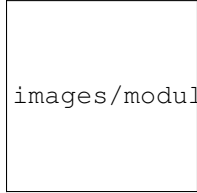
### CartoMap Basic

This is the form snippet from [basic Tab in the CartoMap Wave](#).



images/modules/stay\_CartoDataSet\_messages.png

Fig. 63: Rendered Form CartoDataSet Wave Tab messages

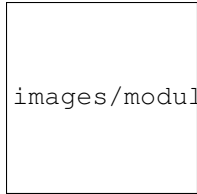


images/modules/stay\_CartoMap\_basic.png

Fig. 64: Rendered Form CartoMap Wave Tab basic

### CartoMap Enrollment

This is the form snippet from [enrollment Tab](#) in the [CartoMap Wave](#).



images/modules/stay\_CartoMap\_enrollment.png

Fig. 65: Rendered Form CartoMap Wave Tab enrollment

### CartoMap Task

This is the form snippet from [task Tab](#) in the [CartoMap Wave](#).

### CartoMap Expiration

This is the form snippet from [expiration Tab](#) in the [CartoMap Wave](#).

### CartoMap Messages

This is the form snippet from [messages Tab](#) in the [CartoMap Wave](#).

### CollectedData Integration Types

CollectedData

The purpose of this integration is to provide a channel to collect data.

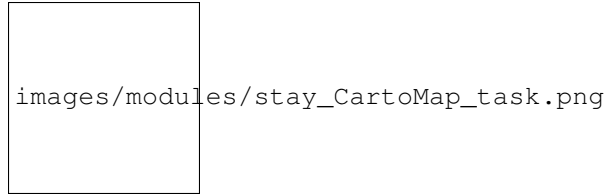


Fig. 66: Rendered Form CartoMap Wave Tab task

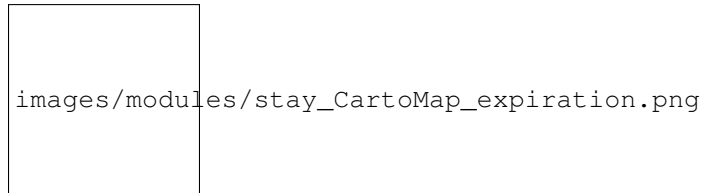


Fig. 67: Rendered Form CartoMap Wave Tab expiration

### CollectedData Wave Configuration

Go through each screen of the Wave Form or Wizard for a CollectedData Wave

#### CollectedData Basic

This is the form snippet from [basic Tab](#) in the [CollectedData Wave](#).

#### CollectedData Enrollment

This is the form snippet from [enrollment Tab](#) in the [CollectedData Wave](#).

#### CollectedData Task

This is the form snippet from [task Tab](#) in the [CollectedData Wave](#).

#### CollectedData Expiration

This is the form snippet from [expiration Tab](#) in the [CollectedData Wave](#).

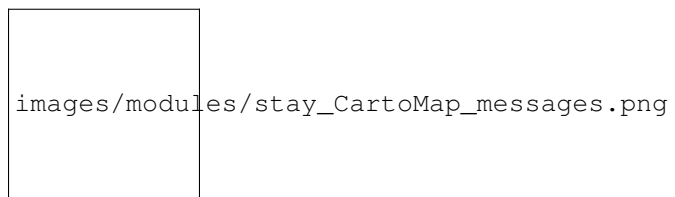


Fig. 68: Rendered Form CartoMap Wave Tab messages

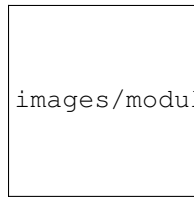


Fig. 69: Rendered Form CollectedData Wave Tab basic



Fig. 70: Rendered Form CollectedData Wave Tab enrollment



Fig. 71: Rendered Form CollectedData Wave Tab task



Fig. 72: Rendered Form CollectedData Wave Tab expiration

## CollectedData Messages

This is the form snippet from [messages Tab](#) in the [CollectedData Wave](#).

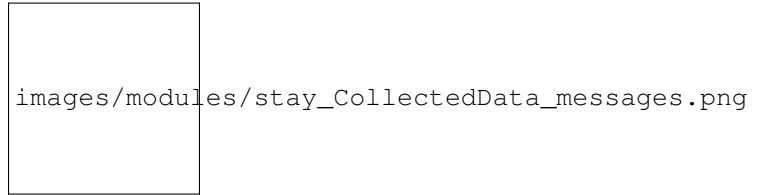


Fig. 73: Rendered Form CollectedData Wave Tab messages

## Csv Integration Types

Csv

The purpose of this integration is to provide a channel to collect data.

### Csv Wave Configuration

Go through each screen of the Wave Form or Wizard for a Csv Wave

### Csv Basic

This is the form snippet from [basic Tab](#) in the [Csv Wave](#).

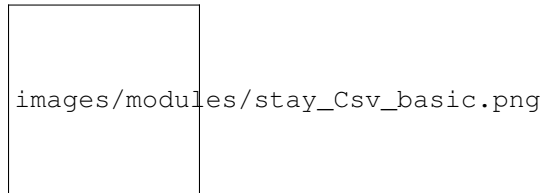


Fig. 74: Rendered Form Csv Wave Tab basic

### Csv Enrollment

This is the form snippet from [enrollment Tab](#) in the [Csv Wave](#).

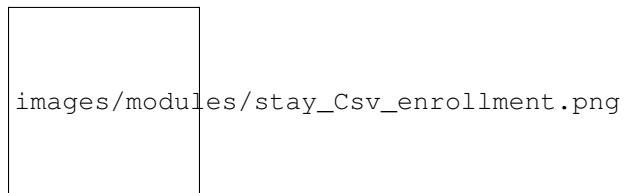


Fig. 75: Rendered Form Csv Wave Tab enrollment

## Csv Task

This is the form snippet from `task` Tab in the `Csv Wave`.

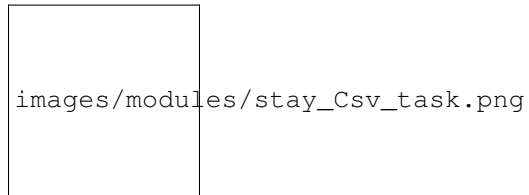


Fig. 76: Rendered Form Csv Wave Tab task

## Csv Expiration

This is the form snippet from `expiration` Tab in the `Csv Wave`.

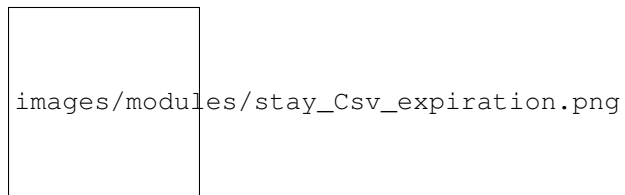


Fig. 77: Rendered Form Csv Wave Tab expiration

## Csv Messages

This is the form snippet from `messages` Tab in the `Csv Wave`.

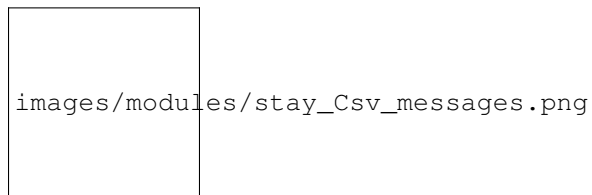


Fig. 78: Rendered Form Csv Wave Tab messages

## Github Integration Types

Github

The purpose of this integration is to provide a channel to collect data.

## Github Wave Configuration

Go through each screen of the Wave Form or Wizard for a Github Wave



## Github Basic

This is the form snippet from `basic` Tab in the Github Wave.

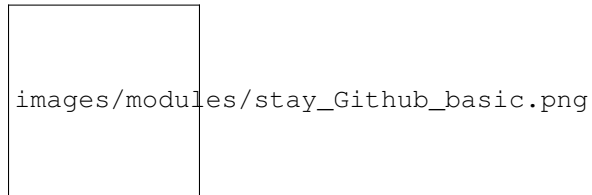


Fig. 79: Rendered Form Github Wave Tab basic

## Github Enrollment

This is the form snippet from `enrollment` Tab in the Github Wave.

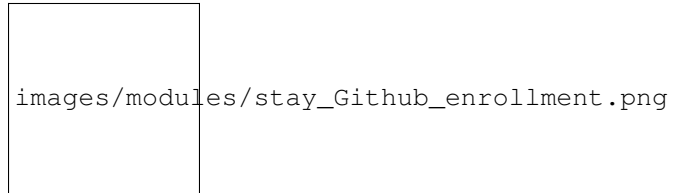


Fig. 80: Rendered Form Github Wave Tab enrollment

## Github Task

This is the form snippet from `task` Tab in the Github Wave.



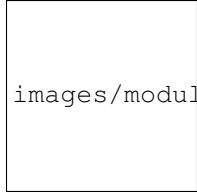
Fig. 81: Rendered Form Github Wave Tab task

## Github Expiration

This is the form snippet from `expiration` Tab in the Github Wave.

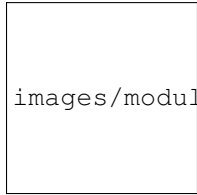
## Github Messages

This is the form snippet from `messages` Tab in the Github Wave.



images/modules/stay\_Github\_expiration.png

Fig. 82: Rendered Form Github Wave Tab expiration



images/modules/stay\_Github\_messages.png

Fig. 83: Rendered Form Github Wave Tab messages

## Mapmob Integration Types

### Mapmob


The purpose of this integration is to provide a channel to collect data.

## Mapmob Wave Configuration

Go through each screen of the Wave Form or Wizard for a Mapmob Wave

### Mapmob Basic

This is the form snippet from [basic Tab in the Mapmob Wave](#).



images/modules/field\_Mapmob\_basic.png

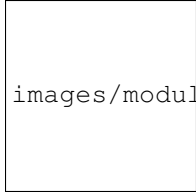
Fig. 84: Rendered Form Mapmob Wave Tab basic

### Mapmob Questions

This is the form snippet from [questions Tab in the Mapmob Wave](#).

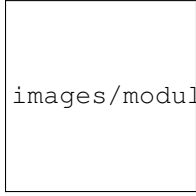
### Mapmob Schedule

This is the form snippet from [schedule Tab in the Mapmob Wave](#).



images/modules/field\_Mapmob\_questions.png

Fig. 85: Rendered Form Mapmob Wave Tab questions

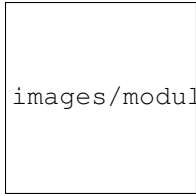


images/modules/field\_Mapmob\_schedule.png

Fig. 86: Rendered Form Mapmob Wave Tab schedule

### Mapmob Enrollment

This is the form snippet from [enrollment Tab](#) in the [Mapmob Wave](#).



images/modules/field\_Mapmob\_enrollment.png

Fig. 87: Rendered Form Mapmob Wave Tab enrollment

### Mapmob Task

This is the form snippet from [task Tab](#) in the [Mapmob Wave](#).

### Mapmob Expiration

This is the form snippet from [expiration Tab](#) in the [Mapmob Wave](#).

### Mapmob Messages

This is the form snippet from [messages Tab](#) in the [Mapmob Wave](#).

## Participants Integration Types

### Participants

The purpose of this integration is to provide a channel to collect data.

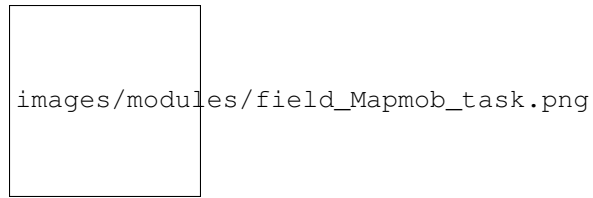


Fig. 88: Rendered Form Mapmob Wave Tab task

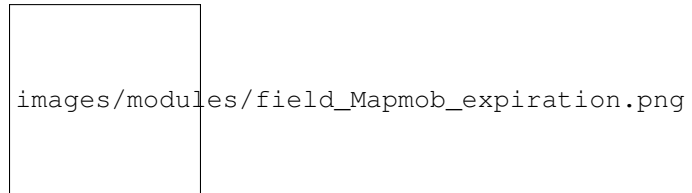


Fig. 89: Rendered Form Mapmob Wave Tab expiration

## Participants Wave Configuration

Go through each screen of the Wave Form or Wizard for a Participants Wave

### Participants Basic

This is the form snippet from [basic Tab in the Participants Wave](#).

### Participants Enrollment

This is the form snippet from [enrollment Tab in the Participants Wave](#).

### Participants Task

This is the form snippet from [task Tab in the Participants Wave](#).

### Participants Expiration

This is the form snippet from [expiration Tab in the Participants Wave](#).

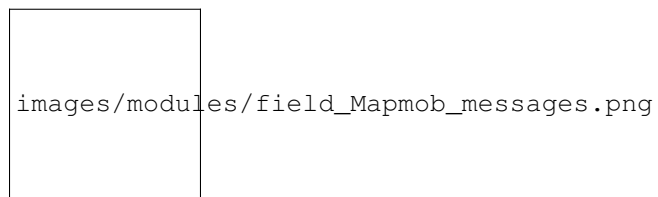


Fig. 90: Rendered Form Mapmob Wave Tab messages

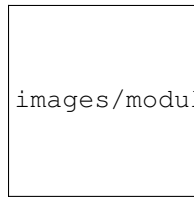


Fig. 91: Rendered Form Participants Wave Tab basic



Fig. 92: Rendered Form Participants Wave Tab enrollment



Fig. 93: Rendered Form Participants Wave Tab task



Fig. 94: Rendered Form Participants Wave Tab expiration

## Participants Messages

This is the form snippet from [messages Tab in the Participants Wave](#).

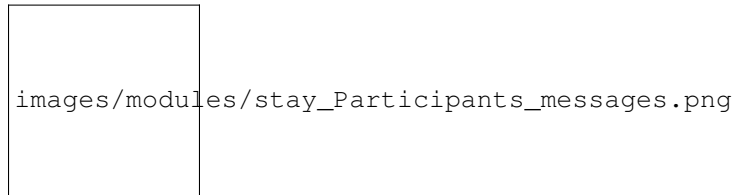


Fig. 95: Rendered Form Participants Wave Tab messages

## PaypalRest Integration Types

PaypalRest

The purpose of this integration is to provide a channel to collect data.

## PaypalRest Wave Configuration

Go through each screen of the Wave Form or Wizard for a PaypalRest Wave

## PaypalRest Basic

This is the form snippet from [basic Tab in the PaypalRest Wave](#).

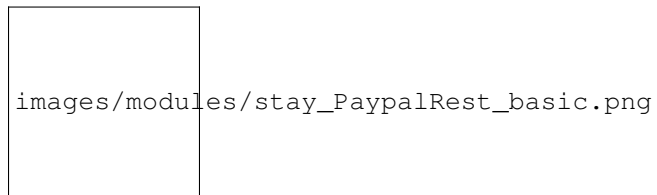


Fig. 96: Rendered Form PaypalRest Wave Tab basic

## PaypalRest Enrollment

This is the form snippet from [enrollment Tab in the PaypalRest Wave](#).

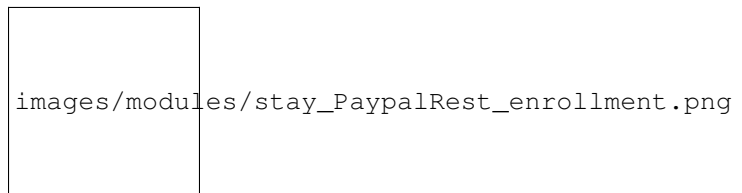


Fig. 97: Rendered Form PaypalRest Wave Tab enrollment

## PaypalRest Task

This is the form snippet from `task` Tab in the `PaypalRest` Wave.

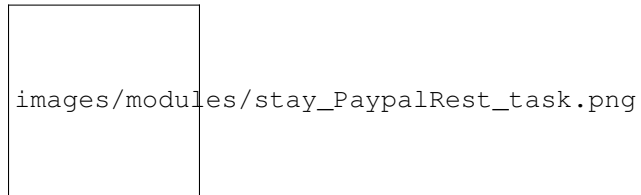


Fig. 98: Rendered Form PaypalRest Wave Tab task

## PaypalRest Expiration

This is the form snippet from `expiration` Tab in the `PaypalRest` Wave.

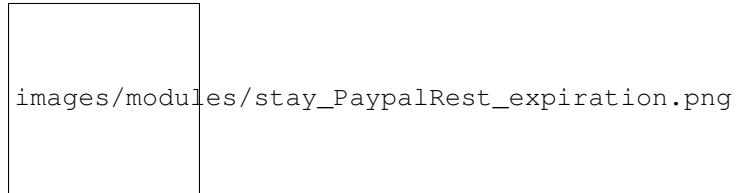


Fig. 99: Rendered Form PaypalRest Wave Tab expiration

## PaypalRest Messages

This is the form snippet from `messages` Tab in the `PaypalRest` Wave.

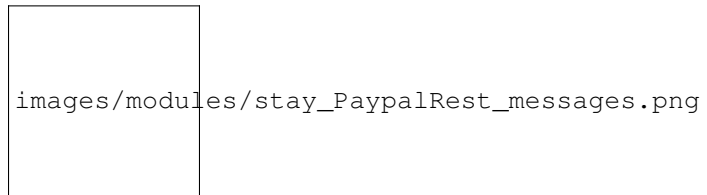


Fig. 100: Rendered Form PaypalRest Wave Tab messages

## RabbitQueue Integration Types

### RabbitQueue

The purpose of this integration is to provide a channel to collect data.

### RabbitQueue Wave Configuration

Go through each screen of the Wave Form or Wizard for a RabbitQueue Wave

## RabbitQueue Basic

This is the form snippet from [basic Tab](#) in the RabbitQueue Wave.

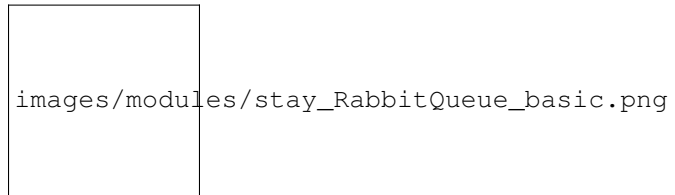


Fig. 101: Rendered Form RabbitQueue Wave Tab basic

## RabbitQueue Enrollment

This is the form snippet from [enrollment Tab](#) in the RabbitQueue Wave.

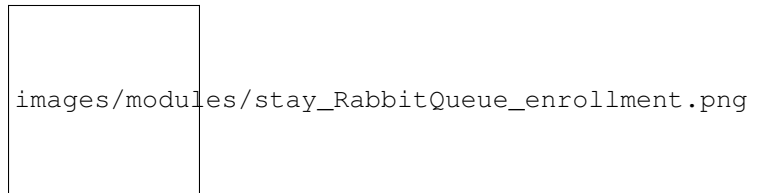


Fig. 102: Rendered Form RabbitQueue Wave Tab enrollment

## RabbitQueue Task

This is the form snippet from [task Tab](#) in the RabbitQueue Wave.

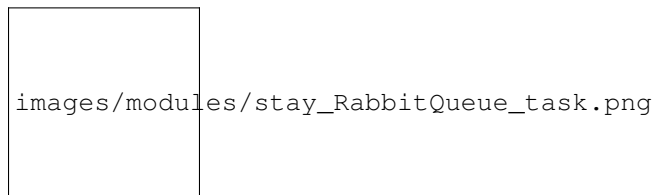


Fig. 103: Rendered Form RabbitQueue Wave Tab task

## RabbitQueue Expiration

This is the form snippet from [expiration Tab](#) in the RabbitQueue Wave.

## RabbitQueue Messages

This is the form snippet from [messages Tab](#) in the RabbitQueue Wave.



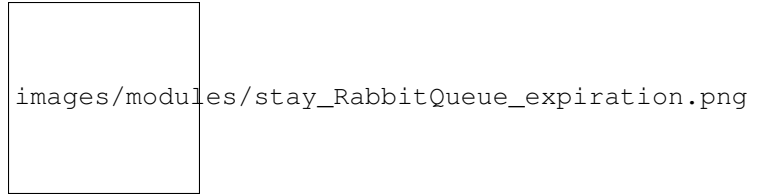


Fig. 104: Rendered Form RabbitQueue Wave Tab expiration

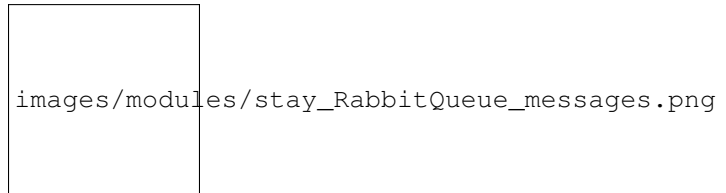


Fig. 105: Rendered Form RabbitQueue Wave Tab messages

## RapidPro Integration Types

### RapidPro

The purpose of this integration is to provide a channel to collect data.

### RapidPro Wave Configuration

Go through each screen of the Wave Form or Wizard for a RapidPro Wave

### RapidPro Basic

This is the form snippet from [basic Tab in the RapidPro Wave](#).

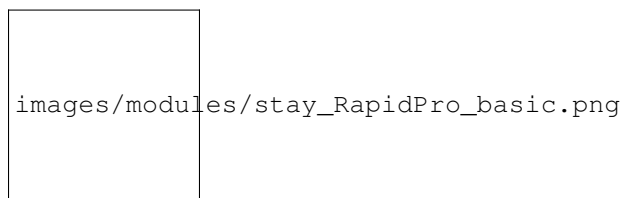


Fig. 106: Rendered Form RapidPro Wave Tab basic

### RapidPro Enrollment

This is the form snippet from [enrollment Tab in the RapidPro Wave](#).

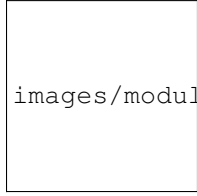
### RapidPro Task

This is the form snippet from [task Tab in the RapidPro Wave](#).



images/modules/stay\_RapidPro\_enrollment.png

Fig. 107: Rendered Form RapidPro Wave Tab enrollment



images/modules/stay\_RapidPro\_task.png

Fig. 108: Rendered Form RapidPro Wave Tab task

## RapidPro Expiration

This is the form snippet from [expiration Tab in the RapidPro Wave](#).



images/modules/stay\_RapidPro\_expiration.png

Fig. 109: Rendered Form RapidPro Wave Tab expiration

## RapidPro Messages

This is the form snippet from [messages Tab in the RapidPro Wave](#).

## SqsQueue Integration Types

SqsQueue

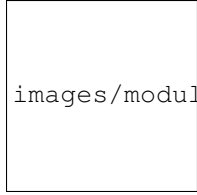
The purpose of this integration is to provide a channel to collect data.

## SqsQueue Wave Configuration

Go through each screen of the Wave Form or Wizard for a SqsQueue Wave

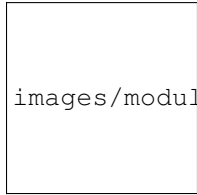
## SqsQueue Basic

This is the form snippet from [basic Tab in the SqsQueue Wave](#).



images/modules/stay\_RapidPro\_messages.png

Fig. 110: Rendered Form RapidPro Wave Tab messages



images/modules/stay\_SqsQueue\_basic.png

Fig. 111: Rendered Form SqsQueue Wave Tab basic

### SqsQueue Enrollment

This is the form snippet from [enrollment Tab](#) in the [SqsQueue Wave](#).



images/modules/stay\_SqsQueue\_enrollment.png

Fig. 112: Rendered Form SqsQueue Wave Tab enrollment

### SqsQueue Task

This is the form snippet from [task Tab](#) in the [SqsQueue Wave](#).

### SqsQueue Expiration

This is the form snippet from [expiration Tab](#) in the [SqsQueue Wave](#).

### SqsQueue Messages

This is the form snippet from [messages Tab](#) in the [SqsQueue Wave](#).

### SubmittedAssignments Integration Types

SubmittedAssignments

The purpose of this integration is to provide a channel to collect data.

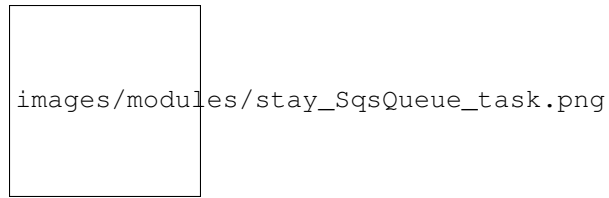


Fig. 113: Rendered Form SqsQueue Wave Tab task

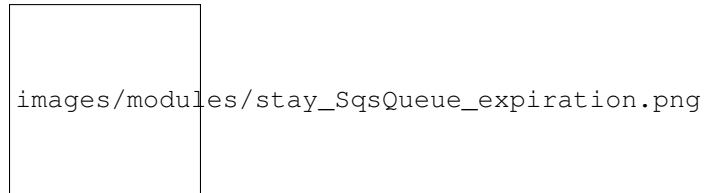


Fig. 114: Rendered Form SqsQueue Wave Tab expiration

### SubmittedAssignments Wave Configuration

Go through each screen of the Wave Form or Wizard for a SubmittedAssignments Wave

#### SubmittedAssignments Basic

This is the form snippet from [basic Tab](#) in the SubmittedAssignments Wave.

#### SubmittedAssignments Enrollment

This is the form snippet from [enrollment Tab](#) in the SubmittedAssignments Wave.

#### SubmittedAssignments Task

This is the form snippet from [task Tab](#) in the SubmittedAssignments Wave.

#### SubmittedAssignments Expiration

This is the form snippet from [expiration Tab](#) in the SubmittedAssignments Wave.

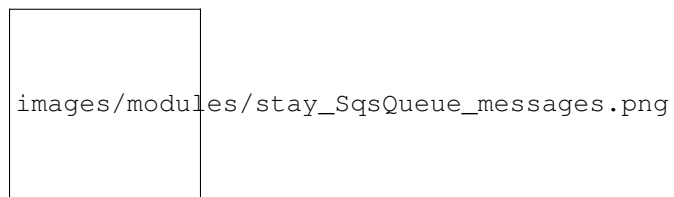
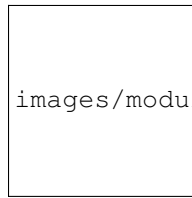


Fig. 115: Rendered Form SqsQueue Wave Tab messages



images/modules/stay\_SubmittedAssignments\_basic.png

Fig. 116: Rendered Form SubmittedAssignments Wave Tab basic



images/modules/stay\_SubmittedAssignments\_enrollment.png

Fig. 117: Rendered Form SubmittedAssignments Wave Tab enrollment



images/modules/stay\_SubmittedAssignments\_task.png

Fig. 118: Rendered Form SubmittedAssignments Wave Tab task



images/modules/stay\_SubmittedAssignments\_expiration.png

Fig. 119: Rendered Form SubmittedAssignments Wave Tab expiration

## SubmittedAssignments Messages

This is the form snippet from [messages Tab in the SubmittedAssignments Wave](#).

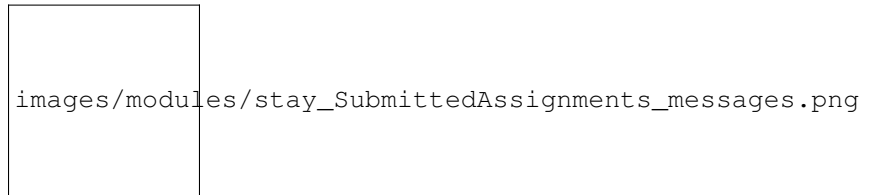


Fig. 120: Rendered Form SubmittedAssignments Wave Tab messages

## Turk Integration Types

Turk

The purpose of this integration is to provide a channel to collect data.

### Turk Wave Configuration

Go through each screen of the Wave Form or Wizard for a Turk Wave

### Turk Basic

This is the form snippet from [basic Tab in the Turk Wave](#).

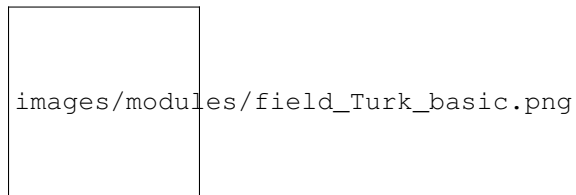


Fig. 121: Rendered Form Turk Wave Tab basic

### Turk Questions

This is the form snippet from [questions Tab in the Turk Wave](#).

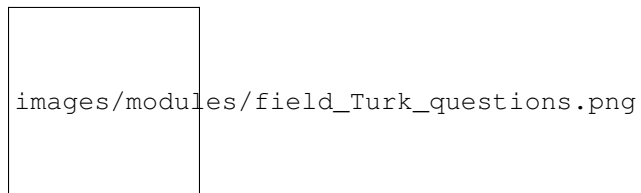


Fig. 122: Rendered Form Turk Wave Tab questions

## Turk Schedule

This is the form snippet from [schedule Tab](#) in the Turk Wave.

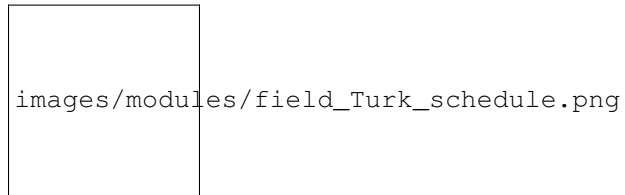


Fig. 123: Rendered Form Turk Wave Tab schedule

## Turk Enrollment

This is the form snippet from [enrollment Tab](#) in the Turk Wave.

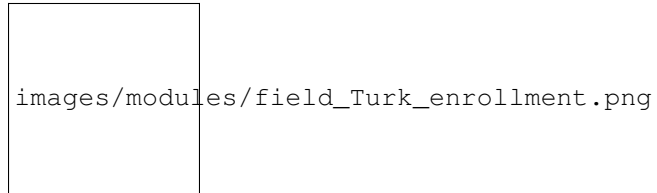


Fig. 124: Rendered Form Turk Wave Tab enrollment

## Turk Task

This is the form snippet from [task Tab](#) in the Turk Wave.

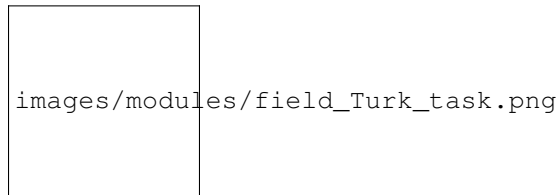


Fig. 125: Rendered Form Turk Wave Tab task

## Turk Qualifications

This is the form snippet from [qualifications Tab](#) in the Turk Wave.

## Turk Expiration

This is the form snippet from [expiration Tab](#) in the Turk Wave.

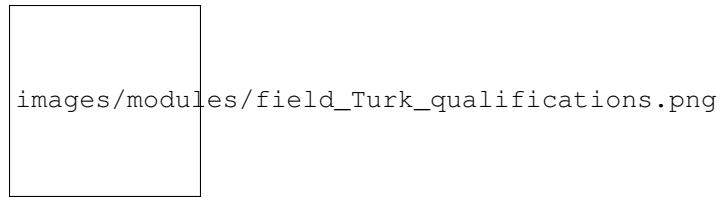


Fig. 126: Rendered Form Turk Wave Tab qualifications

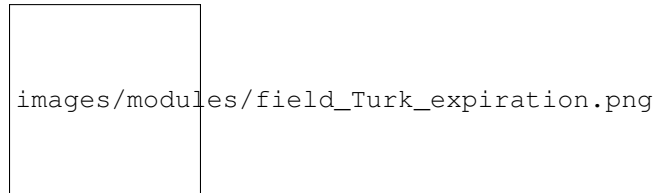


Fig. 127: Rendered Form Turk Wave Tab expiration

## Turk Messages

This is the form snippet from [messages Tab in the Turk Wave](#).

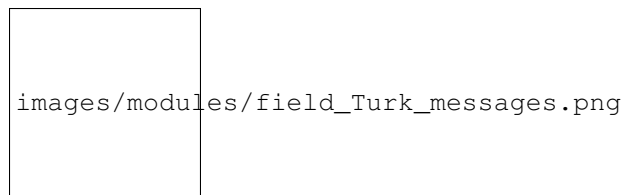


Fig. 128: Rendered Form Turk Wave Tab messages

## TwilioIvr Integration Types

TwilioIvr

The purpose of this integration is to provide a channel to collect data.

### TwilioIvr Wave Configuration

Go through each screen of the Wave Form or Wizard for a TwilioIvr Wave

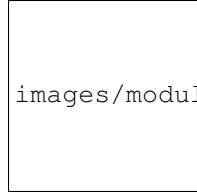
#### TwilioIvr Basic

This is the form snippet from [basic Tab in the TwilioIvr Wave](#).

#### TwilioIvr Enrollment

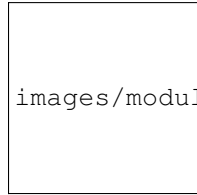
This is the form snippet from [enrollment Tab in the TwilioIvr Wave](#).





images/modules/stay\_TwilioIvr\_basic.png

Fig. 129: Rendered Form TwilioIvr Wave Tab basic

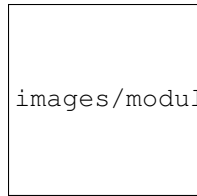


images/modules/stay\_TwilioIvr\_enrollment.png

Fig. 130: Rendered Form TwilioIvr Wave Tab enrollment

### TwilioIvr Task

This is the form snippet from [task Tab](#) in the [TwilioIvr Wave](#).



images/modules/stay\_TwilioIvr\_task.png

Fig. 131: Rendered Form TwilioIvr Wave Tab task

### TwilioIvr Expiration

This is the form snippet from [expiration Tab](#) in the [TwilioIvr Wave](#).

### TwilioIvr Messages

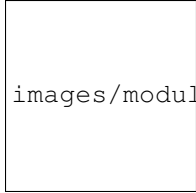
This is the form snippet from [messages Tab](#) in the [TwilioIvr Wave](#).

## 3.6.4 Audit Survey Types

Objective information about external data, like places or images

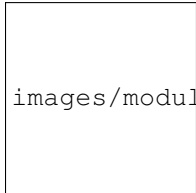
### What's Needed to Deploy

- Qualified Survey Participants
- Images



images/modules/stay\_TwilioIvr\_expiration.png

Fig. 132: Rendered Form TwilioIvr Wave Tab expiration



images/modules/stay\_TwilioIvr\_messages.png

Fig. 133: Rendered Form TwilioIvr Wave Tab messages

## Use Case

When you want to know about individual places from an existing list. Deployed with field workers.

## Integration Types

### Typical Questions

- Go to the 7-11 and 1922 18th St. NW. Take a picture of the front door.
- Is the store open?
- Take pictures of all outdoor ads for sugary drinks.
- Do you see a “We Card” sign?

## Survey Types/Ways to Administer

- Turk, see *Turk Integration Types*
- WebInterview, see *WebInterview Integration Types*
- Mapmob, see *Mapmob Integration Types*
- Api, see *Api Integration Types*
- ObserveWeb, see *ObserveWeb Integration Types*
- JsonResponses, see *JsonResponses Integration Types*

## Survey Participants

Administrators can invite or add participants. Projects with “Allow Self Registration” and an active Screener Wave allow visitors to create an fill out a pre-qualification survey.

## Data Collected

## Technical Notes

Code: field

## 3.7 Users and Participants

A *user* is an individual who has an account on the site. A *participant* is a user who has been enrolled in a mobile job. Users may have one or more *roles*. The most common role is `ROLE_ADMIN`, which gives the user full access when they are logged in.

### 3.7.1 Self-Registration

Users can register themselves on a site. To do so, they simply go to the site, and when prompted to log in, click on “New User Registration”. The first screen prompts them to create a username and password.

The username *must not* contain spaces or punctuation, except for “@” and “.” so that email addresses can be used as usernames.

### 3.7.2 Administrative Registration

Administrators can create users on the site. A user must be created before the person can be enrolled in a specific job. An administrator can create a user by going to the “Users” menu and selecting “Create a New User”.

### Self-Enrollment

A user can register a tracking device as part of their profile. All users must have an account, even if a project is closed and the user is not allowed to log in. This allows a single record for each user so that information like their time zone and mobile phone number do not need to be repeated throughout the project.

Users register themselves by going to the login page

Signing in with Google is trivial, and recommended since Survos Mobile also supports it, making for a very fast login and avoiding yet another account name and password to remember. Alternatively, the user can click on *Create an Account* and be prompted with a screen that asks for email address, username, and password. The email and username must be unique on the site; different users cannot have the same email. The username may be the same as the email, which is what happens if the user registered via Google+.

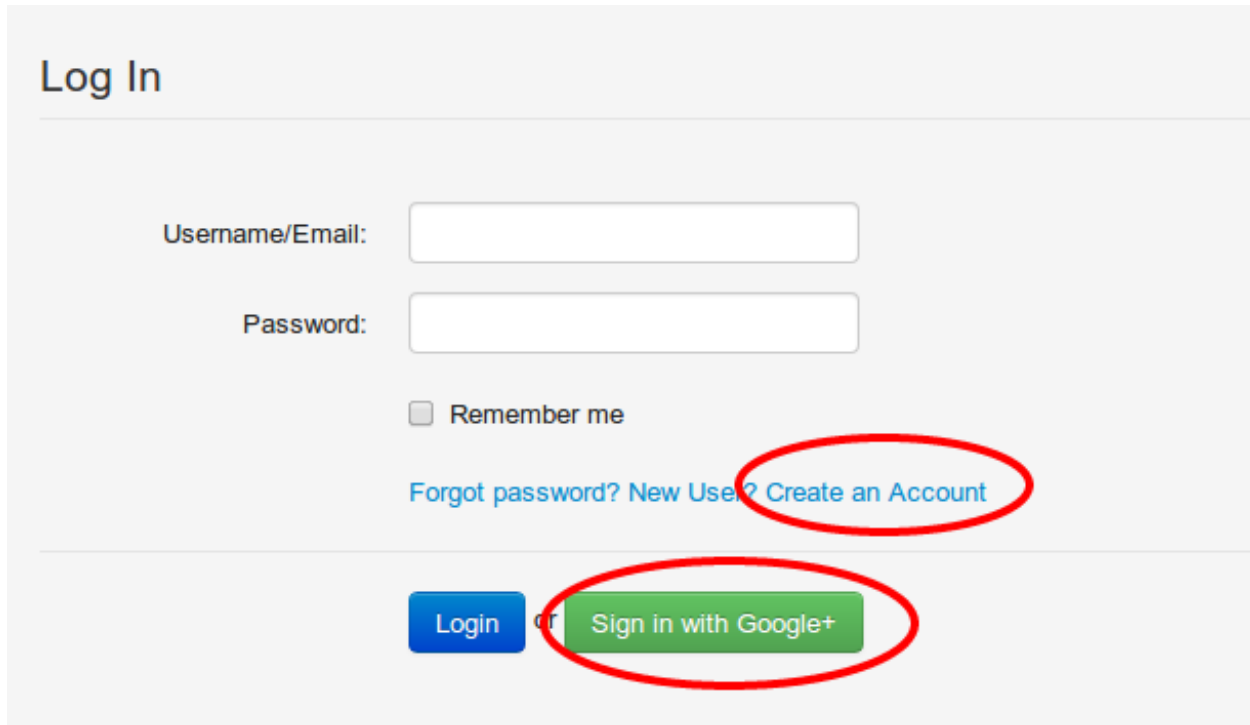
### Profile Editor

After logging in, the user is taken to a page that allows them to complete their profile. The field details are described in the next section, since they are a subset of the fields a project administrator will see.

The user simply fills in the fields and clicks on *Save*.

To enroll a participant:

<https://www.youtube.com/watch?v=2YmuZxtA6vY>



The login screen features a 'Log In' header. Below it are two input fields: 'Username/Email:' and 'Password:'. A 'Remember me' checkbox is located below the password field. A link 'Forgot password? New User? Create an Account' is positioned below the checkbox. At the bottom, there are two buttons: a blue 'Login' button and a green 'Sign in with Google+' button. Both the 'Create an Account' link and the 'Sign in with Google+' button are circled in red.

Fig. 134: Login Screen

## Survos Profile Edit

### Your Survos Profile

profile complete

100%

#### Personal data

Username: \*

tac

Email

tacman@gmail.com

Name

Tac Tacelosky

Required if participant will be using SMS or MMS. Google Voice does NOT work with MMS, it must be an actual carrier-provided number.

Must be the same as the device id set in Traccar or the IMEI if GT500

#### Mobile Surveys

Amazon turk Id

see <https://www.mturk.com/mturk/dashboard>, upper right

Mobile phone

(202) 656-7822

You must be able to send and receive texts (SMS) on this number

Tracking Device ID

18a91e39835f1398

Automatically filled in with Survos Mobile.

Timezone

New York

Cancel

Save

Fig. 135: Survos Profile Editor

## Enrolling Participants

There are three ways participants can be enrolled in a job, self-enrollment, via the “Participants” table, or when a user is created or modified.

### Self-Enrollment

#### Adding Participants to a Job

(TEXT and screenshots here)

#### Enrolling A User from User Profile Page

(text and screenshots)

### Enrolling a Participant

1. Install CHAT app
  - iPhone: [insert instructions]
  - Android: [insert instructions]
2. Add new user to Survos
  - Click “participants” on left-hand column
  - Click “Create new user”
  - Enter their mobile phone #!!!
  - Enter user name (same as baseline ID number)
  - Enter password (same as user name)
  - Click “save”
  - Click “Add new device”
  - Enter their chat number and click on the search button
  - Select “chat-iOS” for type
  - Click “save”
3. Enroll participant
  - Click on the “random” study
  - Click on the “enrollees” tab
  - Click “add new participant”
  - Select the user by clicking on the box to the left of their name/id
  - Enter the start date (if left blank it will start today)
  - “Verbosity” – choose “normal”
  - Click “ADD”

- Go back to the home page
  - Click on “cig” study
  - Click on the “enrollees” tab
  - Click “add new participant”
  - Select the user by clicking on the box to the left of their name/id
  - Enter start date
  - Verbosity – normal
  - Click add
4. Send test survey
- Click on the “random” study
  - Click on the “enrollees” tab
  - Select the box to the left of the participant’s user name
  - Action > Initiate Right now
  - Click “Do action”
5. Schedule survey
- Click on the “cig” study
  - Click on the “enrolles” tab
  - Select the box to the left of the participant’s user name
  - Click on the pencil icon
  - Enter start date
  - Enter number of days: 7
  - Verbosity: normal
  - Frequency: 6
  - Enter start and end time (8 hours)
  - Click submit”
6. Program phone numbers into participants phone
- 202-688-0658 “Cig Survey”
  - 202-601-3699 “Random Survey”
7. Explain incentives
- Give them a copy of the incentives chart

## **3.8 Tutorials**

Contents:

### 3.8.1 Exercise Monitoring

Suppose you wanted to know how much people exercised? You could ask them to keep a diary, but people often forget to fill it out. In this tutorial, we'll set up a nightly text message to ask people a few simple questions about their exercise for the day.

This tutorial shows how to set up a regularly scheduled survey, like a daily diary.

#### Design the Survey

NOTE: Although there's not much typing in this survey, if you want to simply import the survey and skip the typing, read "Importing Surveys" and import the survey called "Exercise 1".

Click on the Surveys menu, and then *Create a New Survey*. You'll be prompted for the code, and the deployment methods. You can deploy the same survey via multiple methods, selecting *how* you will deploy it now will make sure that the survey designer only allows you to create valid questions for your deployment type. For example, you can't create a *camera* question via IVR.

Make the survey code `exercise1`, set the "Deployable via" to "SMS" and click on "Continue". Survey Designer will open.

The first question should be filled out like the image below.



sms transcript



## Adding Questions

To add the first question to a survey, follow these steps:

1. Fill out the question text with **Did you exercise today?** This is a Yes/No question, so choose that as the type from the dropdown menu. Each question has its own code, like the column headers in a spreadsheet or analysis in a statistics program like R or Stata. Type “exercised\_today” as the variable.

We could stop here, if that’s all the data we wanted to collect. But let’s say we’re interested in what kind of exercise, so let’s ask.

2. Add another question by clicking on the ADD QUESTION button. Now fill out the second question with **“What kind of exercise?”**, and make the variable name “exercise\_type”. We could let the user answer whatever they wanted, but this would give us all sorts of variations that would be difficult to analyse. So we’ll restrict the answers by making it a radio question, called that way because like a radio, you can only choose one station at a time. Choose “radio” from the dropdown, and in the Choices, create answers like these:

IMAGE: 1 Running, 2 Team Sports 3: Swimming 4: Yoga 5:Other

We could stop here, but let’s say we’re interested in the relationship between exercise and feeling, so let’s ask.

3. Add another question by clicking on the ADD QUESTION button, and enter “On a scale of 1-5, where 1 is lousy and 5 is great, how do you feel right now?”. Set the type to integer, and to make sure the answers are in the appropriate range, set min to 1 and max to 5.

IMAGE OF QUESTION.

Let’s do a quick look

4. Publish the surveys

## Exporting Collected Data

Go to Export

## 3.9 Tracking

The Posse Platform allows the ability to track people’s location by associating a tracking ID with each participant. Tracks (timestamped coordinates) are sent to an independent tracking server, Survos Tracker. Individual projects pull tracking data from the participants locally, as needed.

This section will review various methods to initiate tracking, and how to integrate tracking with the Posse Platform.

### 3.9.1 GT500 Devices

The GT500 devices are programmed either with the Quealink software and USB cable provided to device administrators or remotely via SMS. The device should be programmed to send the data to the same address as Traccar—that is:

Setting	Value
Server address	tracker.survos.com
Server port	5000

The unique identifier for GT500 devices is always the IMEI number, a string of 15 digits unique to each device. A name can also be programmed, such as *SCTC-105*, but this is simply a tool to make debugging a bit easier.

## 3.9.2 Survos Mobile

Survos Mobile has built-in tracking and currently does not have any configuration options.

## 3.9.3 Traccar

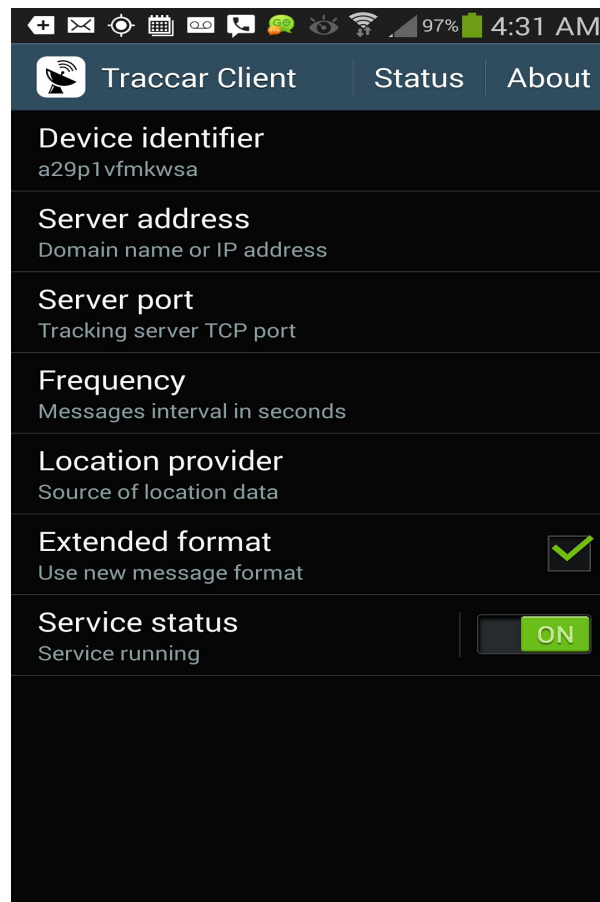
Traccar is a free and open source GPS tracking system. Tracking apps are available for Android and iOS that integrate with the Traccar server or any server that can understand their protocol. Since the Survos Tracker server understands that protocol, a user simply needs to download the app and configure it to point to the right server.

Since the software is available in the app stores, installation is very easy:

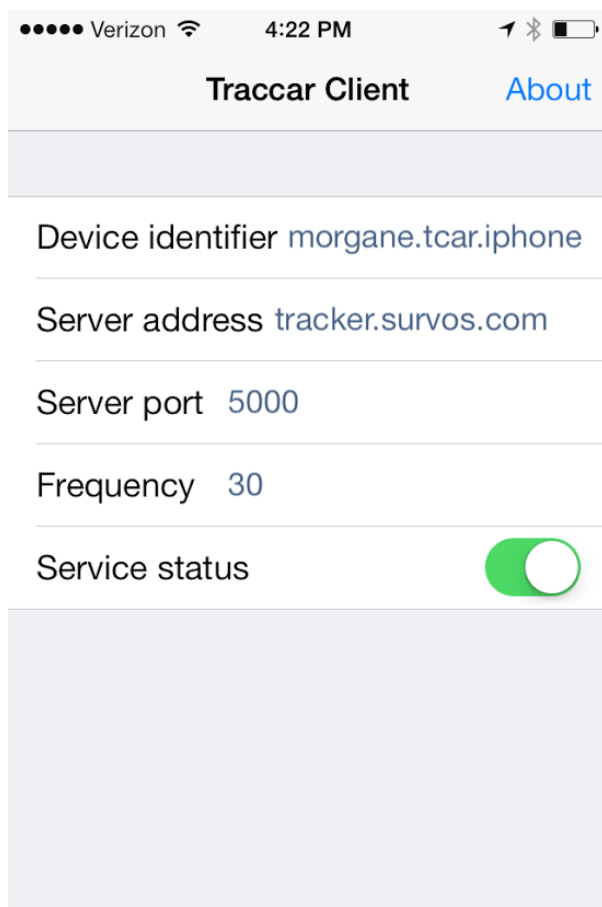
- Android: <https://play.google.com/store/apps/details?id=org.traccar.client> for remote install.
- Android or iOS: Open the App Store on your device, search for *Traccar Client* and install.

### Traccar Configuration

Once you've opened the application, Simply tap on each setting to be prompted for the new value. Use these values to point your device to Survos Tracker. The initial version of Traccar for iOS does not include Location Provider or Extended Format, although the author has indicated that these may be added in the future.



Please note that Apple now prevents apps that use the location service from automatically starting when the phone is powered on. Therefore participants are responsible for turning the app when they boot their phone up, or (even better) keeping their phone charged so it never goes off. Because location apps are a battery drain, this is often difficult. You



can use the Posse Platform to schedule nightly “surveys” that remind people to plug their devices in and ask them to report any problems.

### 3.9.4 Integration with the Posse Platform

Survos Tracker simply records a timestamp and point (latitude and longitude) for each device that sends it information. In order to integrate it with the Posse Platform, there must be a link between users and the device. This can be done either by the user, if they are managing their own account, or by a project administrator, if it is a closed project.

A closed project can only be managed by someone with an account set up that allows access to user data on the system. This is usually the administrator’s account.

See *Enrolling Participants*.

### Registering Devices

In order to associate a tracking device with a user, use the Device panel at the bottom of the profile screen. The GT500’s must be registered with the Master Device Database before those devices can be added to a user profile. The GT500’s are registered at the American Legacy Foundation before they are shipped to partners. Traccar devices will register themselves the first time a point is received by the Survos Tracking Server.

#### Survos Profile Edit

#### Your Survos Profile

profile complete

100%

Personal data

Username: \*

tac

Email

tacman@gmail.com

Name

Tac Tacosky

Mobile Surveys

Amazon turk id

see <https://www.mturk.com/mturk/dashboard>, upper right

Mobile phone

(202) 656-7822

You must be able to send and receive texts (SMS) on this number

Tracking Device ID

18a91e39835f1398

Automatically filled in with Survos Mobile.

Timezone

New York

Required if participant will be using SMS or MMS. Google Voice does NOT work with MMS, it must be an actual carrier-provided number.

Must be the same as the device id set in Traccar or the IMEI if GT500

Cancel

Save

Fig. 136: Survos Device Editor

## Closed Project Enrollment

Projects may be completely closed, (not allowing users to self-enroll or access their data), or they can allow for some user participation. In particular, a project might allow a user to sign up online or in a waiting area before their initial interview. This will allow the user to type in their own name and cell phone number, and then the administrator can simply verify it and add the tracking device as necessary.

If the project is completely closed, then the administrator needs to create a user record for the participant. To do that, simply click on the *Create a New User* link.

The screenshot shows the 'Create A New User' form in the Survos application. The left sidebar contains a list of navigation items, with 'Create A New User' highlighted in blue and a red arrow pointing to it. The main form is titled 'Create A New User' and is divided into two sections: 'Personal data' and 'Survey data'.

**Personal data section:**

- Username: \* (text input)
- Email (text input)
- Name (text input)
- User roles (dropdown menu with 'Select Some Options')
- Password: \* (text input)

**Survey data section:**

- Timezone (dropdown menu with 'New York' selected)
- Amazon turk id (text input)
- Mobile phone (text input)
- Tracking Device ID (text input)

Additional text in the Survey data section:

- see <https://www.mturk.com/mturk/dashboard>, upper right
- You must be able to send and receive texts (SMS) on this number
- Automatically filled in with Survos Mobile.

A 'Save' button is located at the bottom of the form.

Fig. 137: New User Administration

While this screen is fairly self-explanatory, there are a few things worth noting:

- **Username:** This is the identifier that will show up on reports and data exports. You can select something obscure if you want to keep your report data anonymous. You can also choose an email address, since that will be unique. Or depending on the project size, you may choose something like first initial plus last name. There should be no spaces or punctuation in the username (other than “@” and “.” for email addresses). It is simply a code for the user.
- **Name:** Used in some reports, but mostly for administration.
- **Timezone:** Timezone is important with tracking and scheduled surveys, since surveys are often scheduled during waking hours, so the system needs to know the timezone. The default is the timezone where the computer is located, which may be wrong if the participant is being registered remotely, such as over the phone.
- **User Roles:** Gives special permissions to certain users, such as administrative rights.
- **Amazon Turk Id:** If you are paying participants through Amazon Mechanical Turk, this is their account number. User recruitment can also be done via Mechanical Turk, as described in the section “Creating a Recruitment Job” in the “Surveys” chapter.
- **Mobile Phone:** Required to participate in SMS (text) or MMS (text plus photos) mobile surveys. Note that Google Voice will not work with MMS, but does work with SMS surveys.
- **Tracking Device Id:** This is the IMEI (for GT500 devices) or other unique identifier that is being sent to Survos Tracker.

### 3.9.5 Conclusion

User registration is fairly straightforward. It can be done entirely by an administrator, entirely self-administered, or a combination. The user record is a single system-wide way to maintain important information about a single user, such as the email address, timezone, mobile phone, and tracking device.

Once a user is set up in the system, they can participate in mobile surveys.



## CHAPTER 4

---

### Indexes and tables

---

- genindex
- modindex
- search