

Technical Details

Package Name	AkvoDjangoFormGateway
Source Code	https://github.com/akvo/Akvo-DjangoFormGateway
Coverage	https://coveralls.io/github/akvo/Akvo-DjangoFormGateway
Documentation	-
Database Schema	https://dbdocs.io/dedenbangkit/akvo-django-form-gateway
Issues	https://github.com/akvo/Akvo-DjangoFormGateway/issues

Data Module

The data module of the new Django API app for remote data collection using Twilio for WhatsApp includes four main models: Form, Data, Question, and Answer. These models are used to collect, store, and manage the collected data.

Form Model

The Form model is used to define the structure of the data that will be collected. It includes information such as the name of the form, a description of the form, and the fields that will be included in the form.

```
class AkvoGatewayForm(models.Model):
    name = models.CharField(max_length=255)
    description = models.TextField(null=True, default=None)
    version = models.IntegerField(default=1)

    def __str__(self):
        return self.name

    class Meta:
        db_table = "ag_form"
```

Question Model

The children of the Form model is Question model. Each Question includes fields for the ID of the form that the question belongs to, the name of the question, and the type of the question. The type

field is defined using the Question Type enumeration, which includes options such as text, number, photo, geo-location, option and multiple-option.

```
class QuestionTypes:
    geo = 1
    text = 2
    number = 3
    option = 4
    multiple_option = 5
    photo = 6
    date = 7

    FieldStr = {
        geo: 'Geo',
        text: 'Text',
        number: 'Number',
        option: 'Option',
        multiple_option: 'Multiple_Option',
        photo: 'Photo',
        date: 'Date',
    }

class AkvoGatewayQuestion(models.Model):
    form = models.ForeignKey(
        to=AkvoGatewayForm,
        on_delete=models.CASCADE,
        related_name="ag_form_questions",
    )
    order = models.BigIntegerField(null=True, default=None)
    text = models.TextField()
    type = models.IntegerField(choices=QuestionTypes.FieldStr.items())
    required = models.BooleanField(null=True, default=True)

    def __str__(self):
        return self.text

    class Meta:
        db_table = "ag_question"
```

```

class AkvoGatewayQuestionOption(models.Model):
    question = models.ForeignKey(
        to=AkvoGatewayQuestion,
        on_delete=models.CASCADE,
        related_name="ag_question_question_options",
    )
    order = models.BigIntegerField(null=True, default=None)
    code = models.CharField(max_length=255, default=None, null=True)
    name = models.TextField()

    def __str__(self):
        return self.name

    class Meta:
        db_table = "ag_option"

```

Data Model

The Data model is used to store the collected data. It includes fields for the ID of the form that the data belongs to, the name of the data, and the submitter (phone number), as well as metadata such as the date and time that the data was collected and the location of the data collection.

```

class StatusTypes:
    draft = 1
    submitted = 2

class AkvoGatewayData(models.Model):
    name = models.TextField()
    form = models.ForeignKey(
        to=AkvoGatewayForm,
        on_delete=models.CASCADE,
        related_name="ag_form_data",
    )
    geo = models.JSONField(null=True, default=None)
    phone = models.CharField(max_length=25)
    status = models.IntegerField(default=StatusTypes.draft)
    created = models.DateTimeField(auto_now_add=True)
    updated = models.DateTimeField(default=None, null=True)

    def __str__(self):

```

```
return self.name
```

```
class Meta:
    ordering = ["id"]
    db_table = "ag_data"
```

Answer Model

The Answer model is used to store the answers to the questions in the Form model. It includes fields for the ID of the answer, the ID of the form data, the ID of the question, and the answer itself. The Answer model is also used to store the status of the submitter session. When the submitter completes all the question in the session, the data model should be marked as done.

```
class AkvoGatewayAnswer(models.Model):
    data = models.ForeignKey(
        to=AkvoGatewayData,
        on_delete=models.CASCADE,
        related_name="ag_data_answer",
    )
    question = models.ForeignKey(
        to=AkvoGatewayQuestion,
        on_delete=models.CASCADE,
        related_name="ag_question_answer",
    )
    name = models.TextField(null=True, default=None)
    value = models.FloatField(null=True, default=None)
    options = models.JSONField(default=None, null=True)

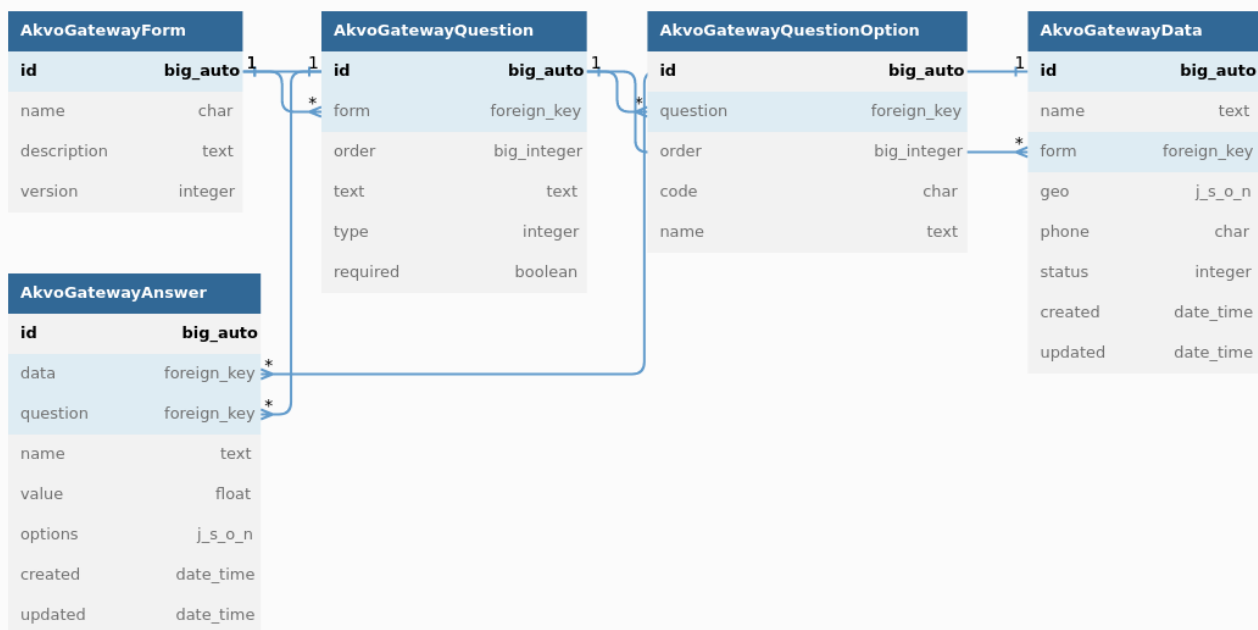
    created = models.DateTimeField(auto_now_add=True)
    updated = models.DateTimeField(default=None, null=True)

    def __str__(self):
        return self.data.name

    class Meta:
        db_table = "ag_answer"
```

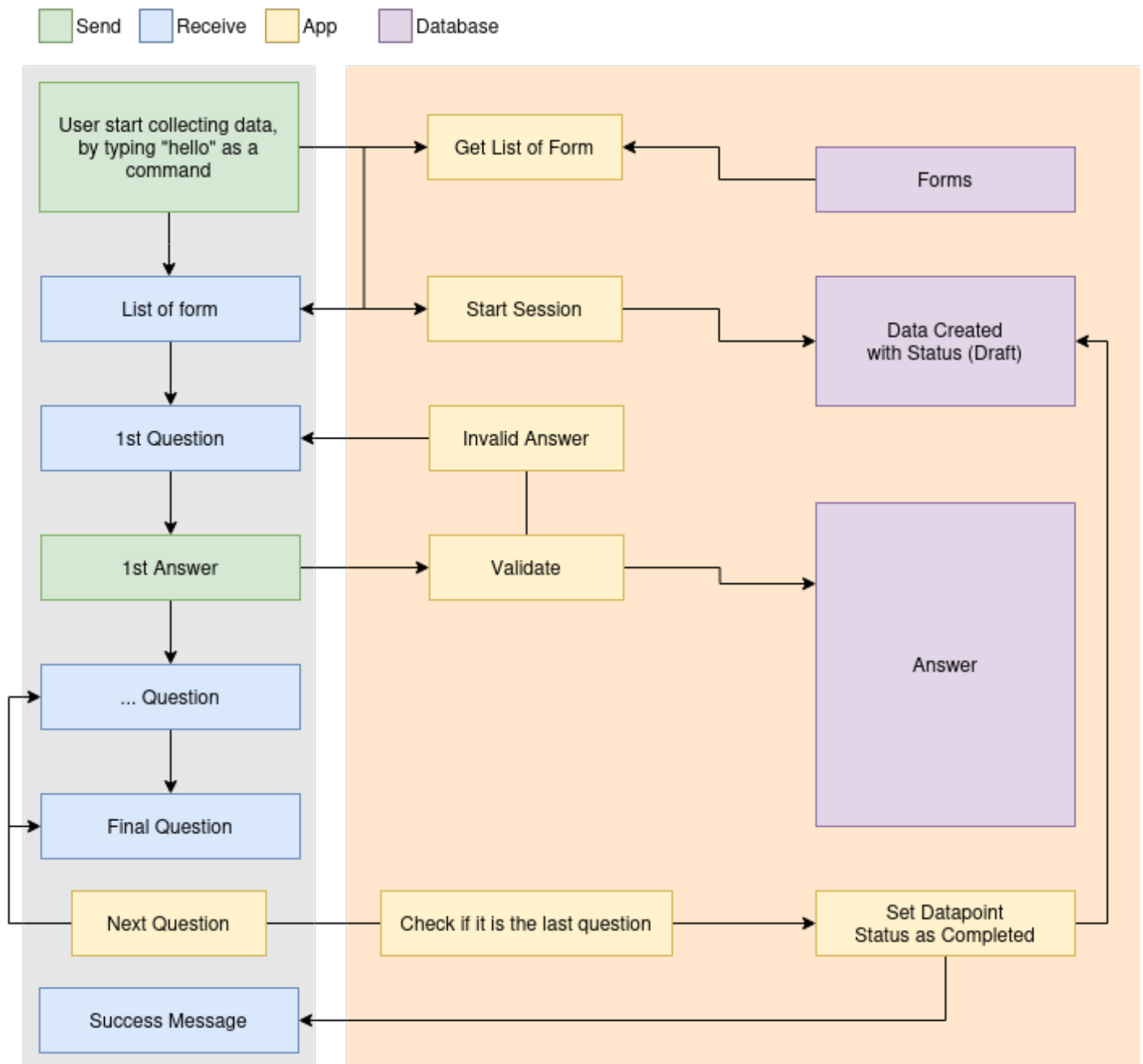
Database Schema

Overall here is the database schema from the Data Models



Workflow

The workflow image below illustrates the process.



1. At first the Django API APP receives incoming messages and checks if the message contains pre-configured parameter which linked to a particular form, e.g when user type "*complain*"; there will be a form in which will then be initiated for the user. This also initiates a data record and sets status as draft
2. Retrieve the questions for the selected form and start asking the user the questions one by one. Once the user answers a question, validate the answer. If the answer is invalid, notify the user that their answer was wrong and prompt them to answer again. If the answer is valid, store the answer in the **Answer model** and move on to the next question.
3. Once all questions have been answered, the **draft** status in the **Data record** changed to **completed**. Still Draft