

Programming Boot Camp

# Adaloでのデータ操作と外部連携

東京工業大学 2021/11/6

Ryo Imahashi

# 目次

- 前回のふりかえりと今回のゴールの確認
- データベースについて学ぼう
- データベース設計
- データベース操作
- サンプルアプリを改善しよう
- 外部サービス連携
- 演習
- まとめ

# 前回のふりかえりと今回のゴールの確認

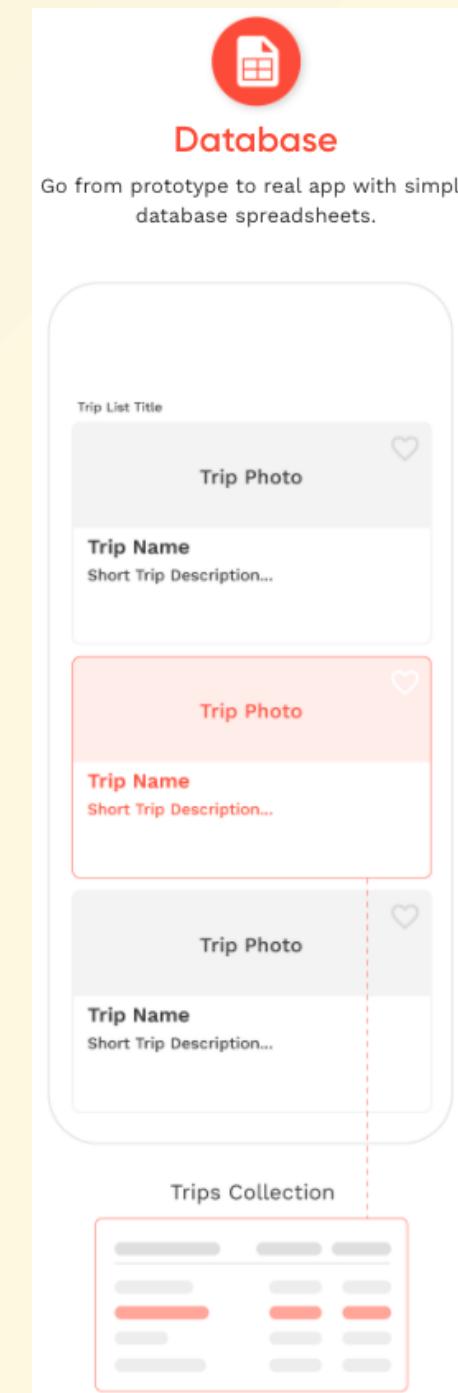
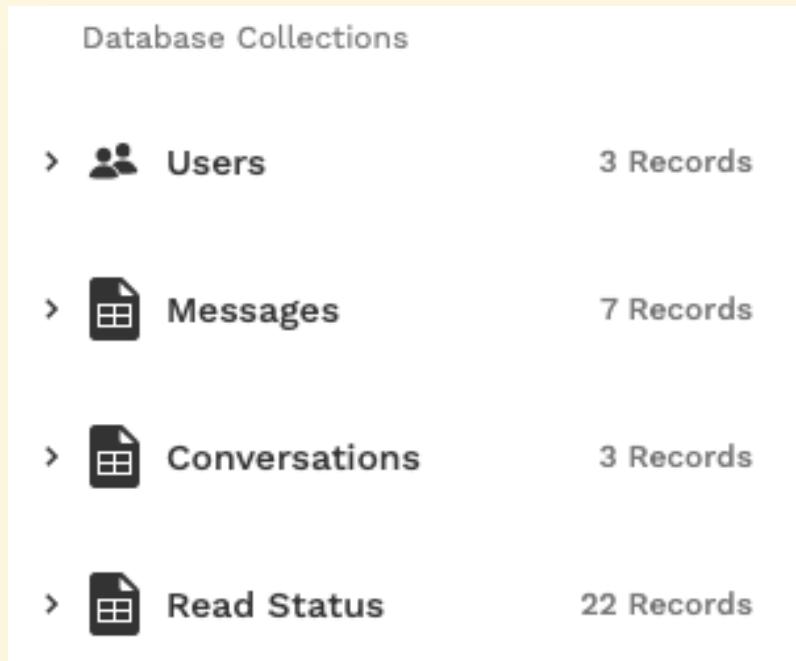
- 前回のレクチャーでは、ノーコードツールのAdaloについて紹介し、ペットの健康管理アプリを題材にアプリのUIを作成しました。
  - レクチャーでは、データベースを必要としない、シンプルなコンポーネントを使いました。(演習の中でデータベースを使った人もいるかもしれません)
- 今回のレクチャーでは、引き続きAdaloを使って、前回作ったUIに合わせたデータベースを構築し、アプリからデータを操作できるようにしていきましょう。
- その後は、Adaloだけでは実現できないことを、Adaloと外部のサービスを連携させることで実現する方法をいくつかご紹介します。

# データベースについて学ぼう

まず、これから学んでいくデータベースがどのようなものかを確認します。

# Database(前回の復習)

- 整理されたデータの集合。
- データの登録、読み込み(表示)、更新、削除が行われる。
- 例: Chatアプリの場合



- データベースとは、電子的にアクセスされる情報の整理された集合です。
- データベースを使ってユーザーはコンピュータに保存されている情報を作成(CREATE)、読み取り(READ)、更新(UPDATE)、削除(DELETE)することができます。これらの機能を総称してCRUDと呼びます。
- データベースは、よく「表計算ソフトのようなもの」と例えられます。(実際に、表計算ソフトをデータベースとして使うことも可能です)

# Adaloのデータベースの基本



このアイコンでAdaloのデータベースにアクセスできます。  
Adaloのデータベースの構成要素は、以下の3つです。

- Collection
- Property
- Record

# Collectionとは

同じ属性(Property)を持ったデータの集まり

The screenshot shows a database interface with a sidebar on the left and a main table view on the right.

**Left Sidebar:**

- + Database Collections
- Database Collections
- Users (selected, 3 Records)
- ⋮
- Email
- Password
- Username
- Full Name
- Profile Picture
- Conversations (Creator)s
- Messages (Sender)s
- Read Statuses
- Conversations (Members)s
- Friends
- + ADD PROPERTY
- Messages (7 Records)
- Conversations (3 Records)

**Main View:**

**Header:** Users

**Actions:** + ADD USER, Up/Down arrows, Cloud icon, Share icon, Search icon

**Table Headers:** Email, Password, Username, Full Name, Profile Picture

**Data Rows:**

	Email	Password	Username	Full Name	Profile Picture
<input type="checkbox"/>	fuga@hoge.com	[hidden]		三人目のユーザー	
<input type="checkbox"/>	hoge@fuga.com	[hidden]		東 工大	
<input type="checkbox"/>	imahashi@example.com	[hidden]		今橋 陵	

**Bottom Right:** DONE

- Collectionは、データベースで扱う様々なデータをデータの種類ごとに分割し、整理するためのものです。(類似の言葉として、「テーブル」があります)
- アプリの中でユーザーが登録、表示、更新、削除といった操作を行うものがCollectionとなります。(「名詞」として表現できるものがCollectionになることが多いと言われます)
- デフォルトでは、UsersがCollectionとして用意されており、それ以外は開発するアプリに合わせて追加していきます。

\* どのようなCollectionを用意するかを決めるのはとても難しいです。練習しながら慣れていきましょう。(悩んだ時は、メンター陣に相談するのもオススメです)

## Propertyとは

- Propertyは、コレクションに含まれる(可能性のある)データが持つ項目です。
- 例えばUsers Collectionでは、Eメール、パスワード、名前、プロフィール写真などのPropertyを持つことが考えられます。

The screenshot shows a MongoDB interface with the following details:

- Collection: Users
- Records: 8 Records
- Properties listed:
  - Email
  - Password
  - Username
  - Full Name
  - Profile Picture
  - Conversations (Creator)s
  - Messages (Sender)s
  - Read Statuses
  - Conversations (Members)s
  - Friends

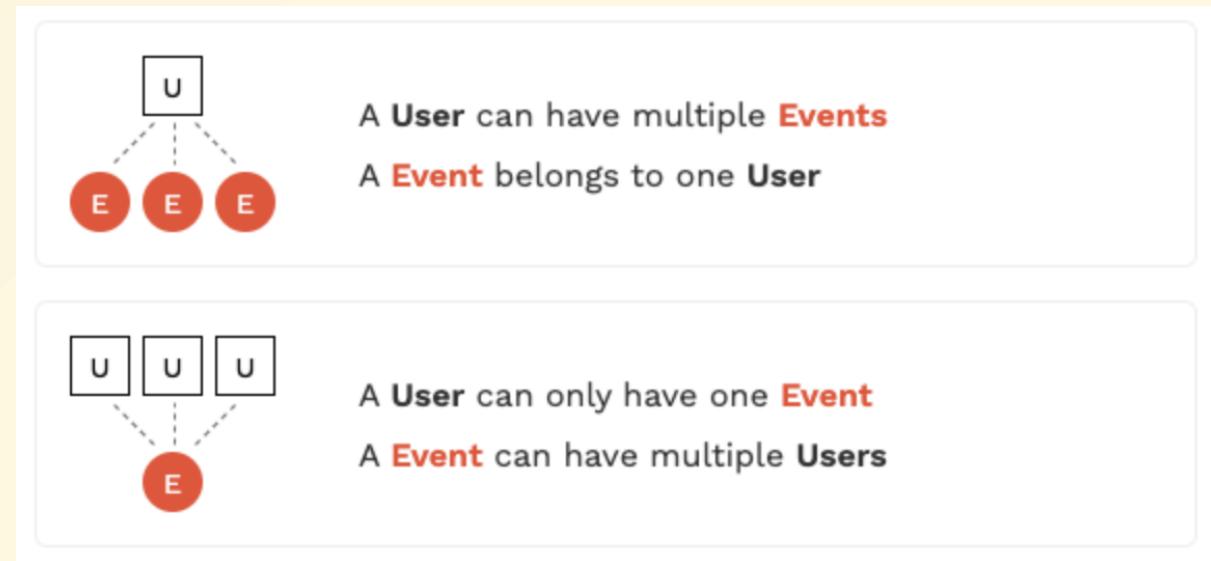
- Propertyとして扱うデータがどのようなものかを定義するため、様々なTypeが用意されています。Propertyの追加時にどれか一つを選択します。
  - Text
  - Number
  - True/False
  - Date/Time
  - Date
  - Image
  - File
  - Relationship

## Relationshipとは

- 1つのレコード(データセット)に対して多数のプロパティを保存する代わりに、Relationshipと呼ばれる複数のコレクションを関連づけるための特別なプロパティを設定します。これにより、Collectionを人間が理解しやすい形で分割することができます。
- AdaloのRelationshipでは、コレクション間でリンクされるレコードの数に応じて、1対多と多対多という2つの種類のいずれかを選択します。

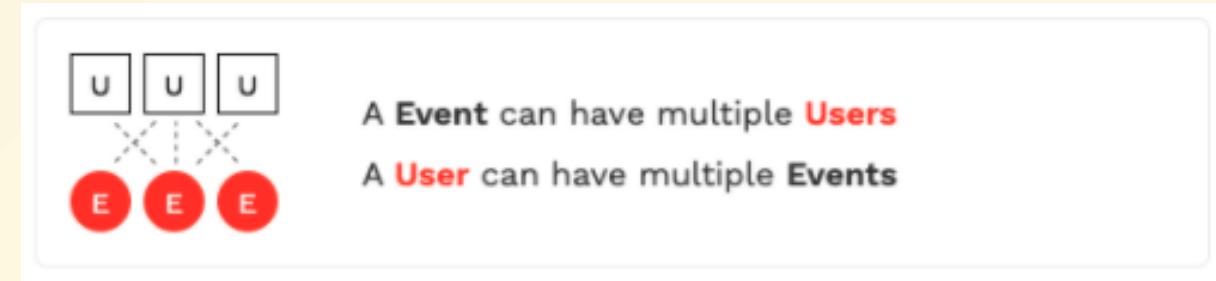
## 1対多

- 1つのレコードが、別のコレクションにある複数のレコードと関係を持つことを意味します。
- 表示しているCollectionを1と多のどちらにするかで、2種類の選択肢が現れます。
- 例えば、主催者がイベントに対して1人だけいる場合の、主催者とイベントのRelationshipは1対多です。



## 多対多

- 両方のコレクションのレコードが、もう一方のコレクションの複数のレコードにリンクできることを意味します。
- 例えば、参加者は複数のイベントに参加できるし、イベントには複数の参加者がいるという場合の、参加者とイベントのRelationshipは多対多です。



## Recordとは

- Recordは、コレクション内へ情報を保存する際に1セットとなるデータの組み合わせです。
- Usersの例では、1人のユーザーがコレクション内の1つのレコードとみなされます。

Users					
	<a href="#">Email</a>	<a href="#">Password</a>	<a href="#">Username</a>	<a href="#">Full Name</a>	<a href="#">Profile Picture</a>
<input type="checkbox"/>	fuga@hoge.com	[hidden]		三人目のユーザー	
<input type="checkbox"/>	hoge@fuga.com	[hidden]		東 工大	
<input type="checkbox"/>	imahashi@example.com	[hidden]		今橋 陵	

DONE

- Recordは基本的にアプリの画面上のフォームから登録できるようになりますが、Recordの表示中に右上の「+Add xxxx」ボタンを押して、右の画像のようなフォームから登録することも可能です。
- Collection内のRecordの検索や、CSVファイルのアップロード(インポート)・ダウンロードも可能です。

New User

Email

Password

Username

Full Name

Profile Picture

Choose image

CANCEL

SAVE

# データベース設計

前回のレクチャーで作成したサンプルアプリのUIを見ながら、保存が必要なデータを考えて、データベースを設計・構築しましょう。

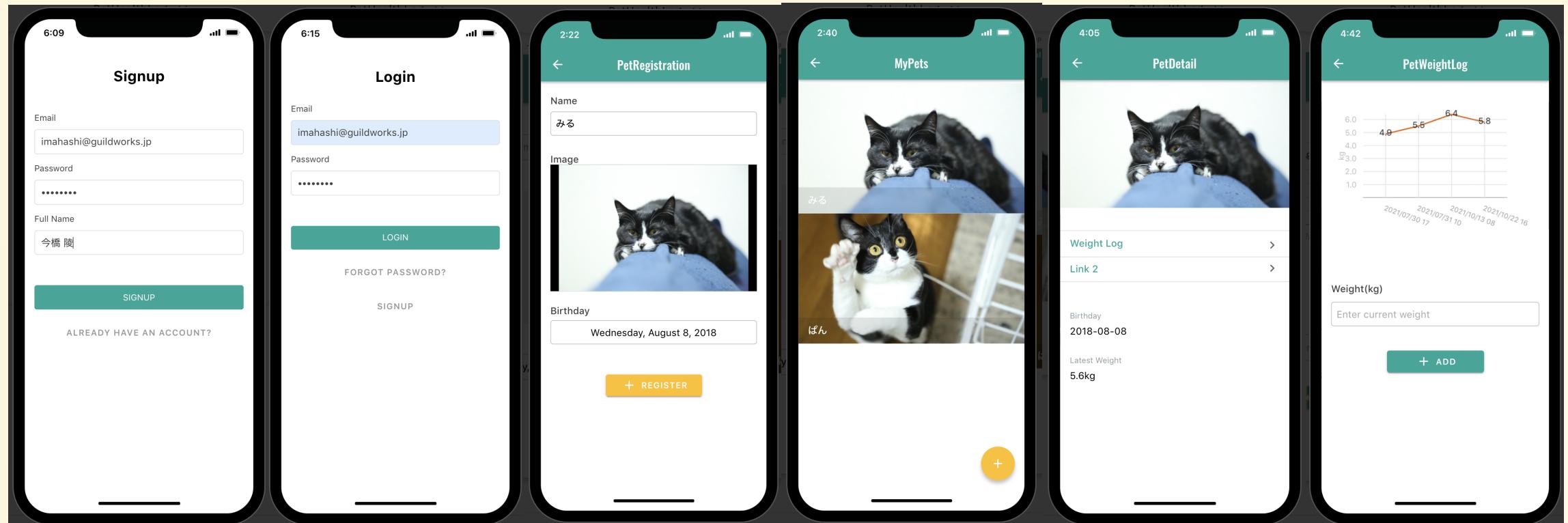
## 前回のレクチャーで作成したアプリのクローン用URL

- 以下のURLからアプリをクローンしてください。それを使ってここからのレクチャーを進めます。

<https://previewer.adalo.com/014fd9d1-80c6-4325-899a-d943e778c865>

# データベースを設計してみよう

サンプルアプリのUIを見ながら、データベースを設計してみましょう。手順は次のページで紹介します。



## データベース設計の手順

1. UIを見ながら、保存が必要になるデータをリストアップしましょう。テキストエディタ(メモ帳アプリ等)に書き起こしてください。
2. リストアップしたデータがどのようなCollectionに分類できるかを考えて、Adaloのデータベースに必要なCollectionを作成しましょう。
3. 1でリストアップしたデータを適切なCollectionにPropertyとして追加してください。その際、適切なTypeを選択してください。
4. 他のCollectionと関連を持つCollectionには、Relationship Propertyを設定しましょう。

\* 次のスライド以降に解説がありますが、答えを見る前に一度自分で手を動かして考えてみることをオススメします。

\* 絶対的な正解はありません。悩んだら、直感に従って進めてみてください。

## 解説

UIを見ながら、保存が必要になるデータをリストアップすると、以下のようになりました。

- ユーザーのEmail
- ユーザーのパスワード
- ユーザーのFullName
- ペットの名前
- ペットの写真
- ペットの誕生日
- ペットの体重
- ペットの体重の登録日時

- その他のデータを挙げられた人がいれば、教えてください！

リストアップしたデータがどのようなCollectionに分類できるかを考えて、今回はこの3つに分類することにします。

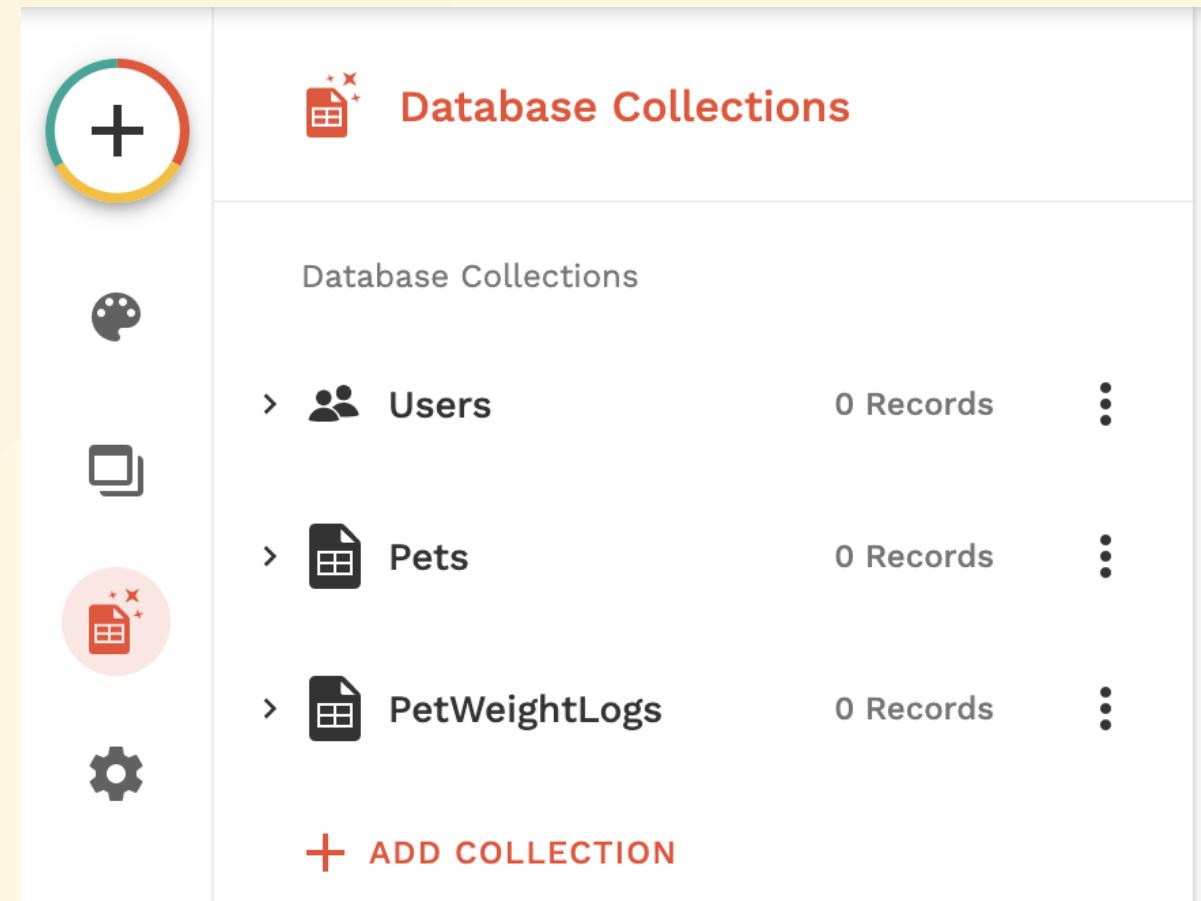
- Users
- Pets
- PetWeightLogs

- ユーザーのCollectionとペットのCollectionの2つを用意した人は多いのではないでしょうか？
- ペットの体重記録のCollectionは用意しなかった人もいるかもしれません。(ペットの体重とその登録日時をペットのCollectionに含める方法も間違いではありません。後ほど解説します。)
- その他のCollectionの分類をした人はいますか？

## Collectionの分類に関する補足

- 「Aが決まればBが1つに決まる」という関係が成り立つ時、AをCollectionに、BをそのCollectionのPropertyにする場合が多いです。
  - ユーザーが決まれば、ユーザーのEmail、パスワード、FullNameがそれぞれ1つに決まります。
  - ペットが決まれば、ペットの名前、写真、誕生日がそれぞれ1つに決まります。
- 「Aに対してBが複数存在する」という関係が成り立つ時、AとBは別々のCollectionに分割することが多いです。
  - (1匹の)ペットに対して、ペットの体重とその登録日時は複数存在します。

- CollectionをAdaloのデータベースに登録しておきます。
- Usersはデフォルトで作成されています。

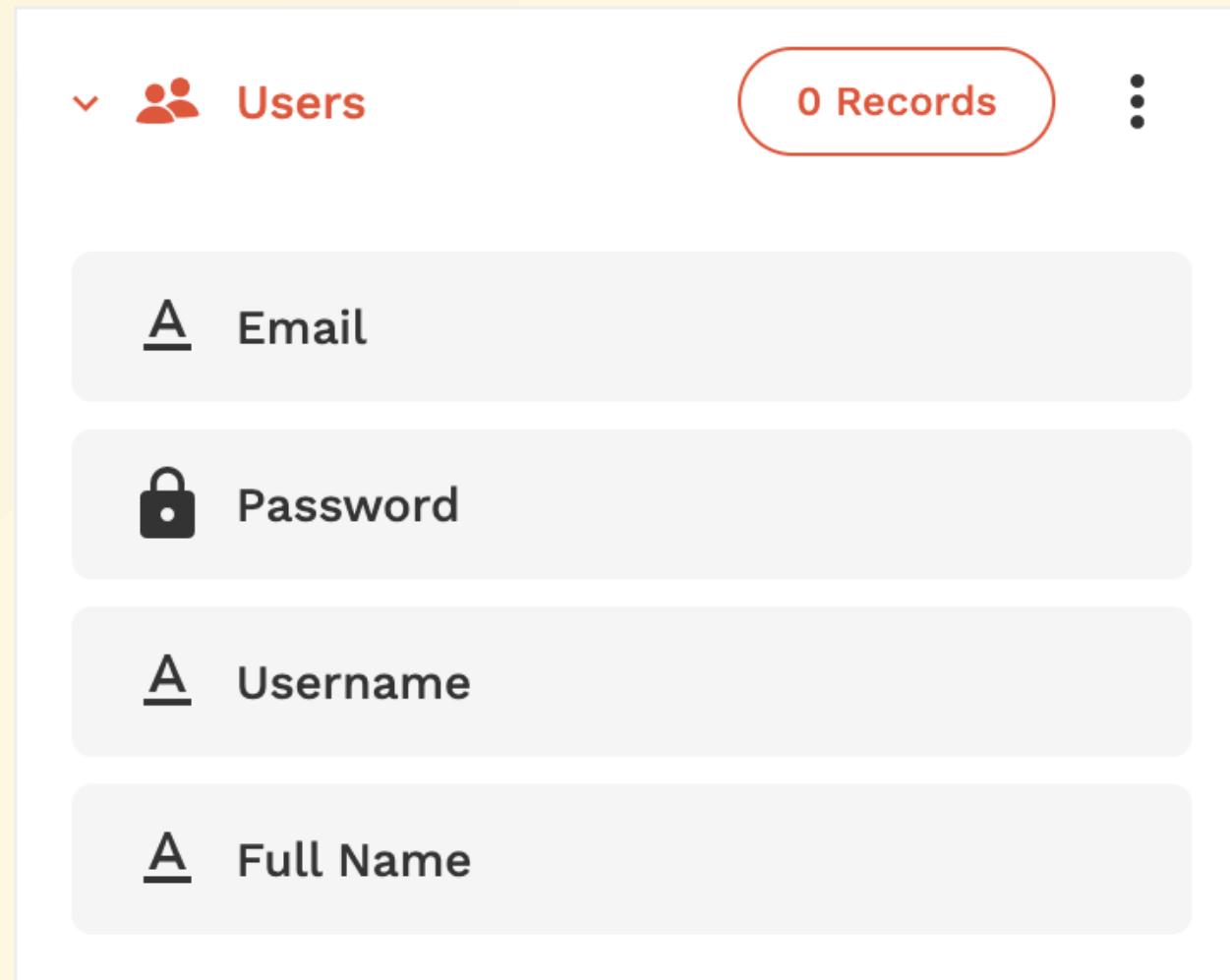


次に、1でリストアップしたデータを適切なCollectionの配下にPropertyとして追記すると、以下のようになりました。()の中は選択するTypeです。

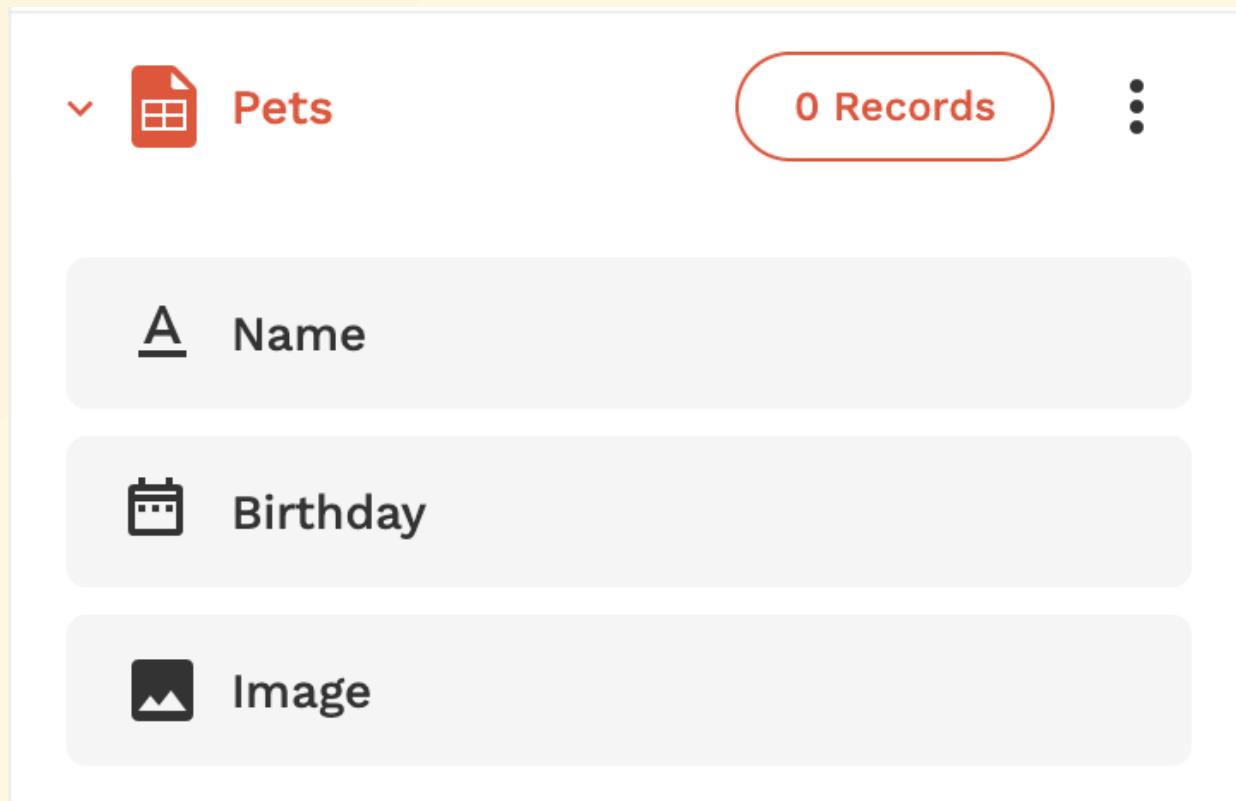
- Users
  - Email(Text)
  - Password(※Password)
  - FullName(Text)
- Pets
  - Name(Text)
  - Image(Image)
  - Birthday(Date)
- PetWeightLogs
  - WeightKg(Number)
  - RegisteredTime(Date&Time)

\* Passwordはデフォルトで設定される特殊なTypeです。

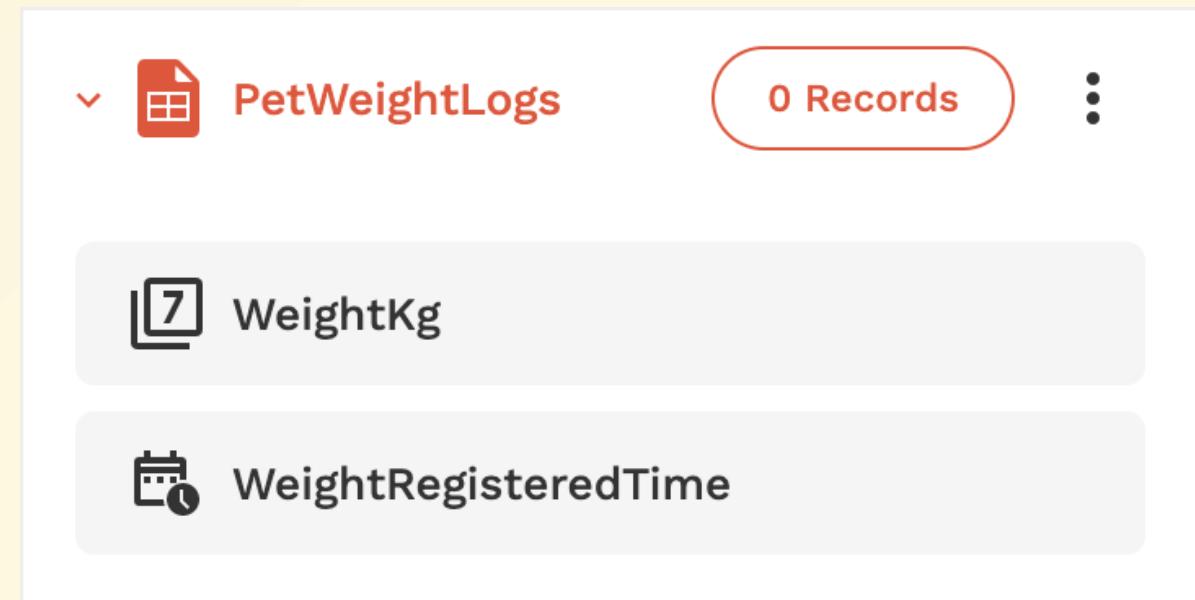
- Adaloで実際にPropertyを追加しましょう。
- Users Collectionはデフォルトで設定済みで、必要な項目は含まれています。
- Usernameは不要ですが、削除できないのでそのままにしておきましょう。



- Pets CollectionのPropertyはこのようになります。

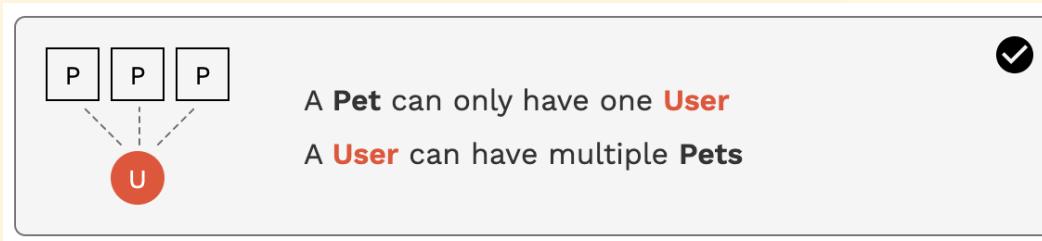


- PetWeightLogs Collection のPropertyはこのようになります。
- Collection追加時にデフォルトで設定されるName Propertyは不要なので、削除しましょう。
  - ドラッグアンドドロップで順番がCollection内の一一番上でなくなるように移動すれば、削除できます。



最後に、他のCollectionと関連を持つCollectionには、Relationship Propertyを設定しましょう。

- Users Collectionを選択して、Pet Collectionとの1対多のRelationshipを追加します。



▼ Users 0 Records ⋮

Email

Password

Username

Full Name

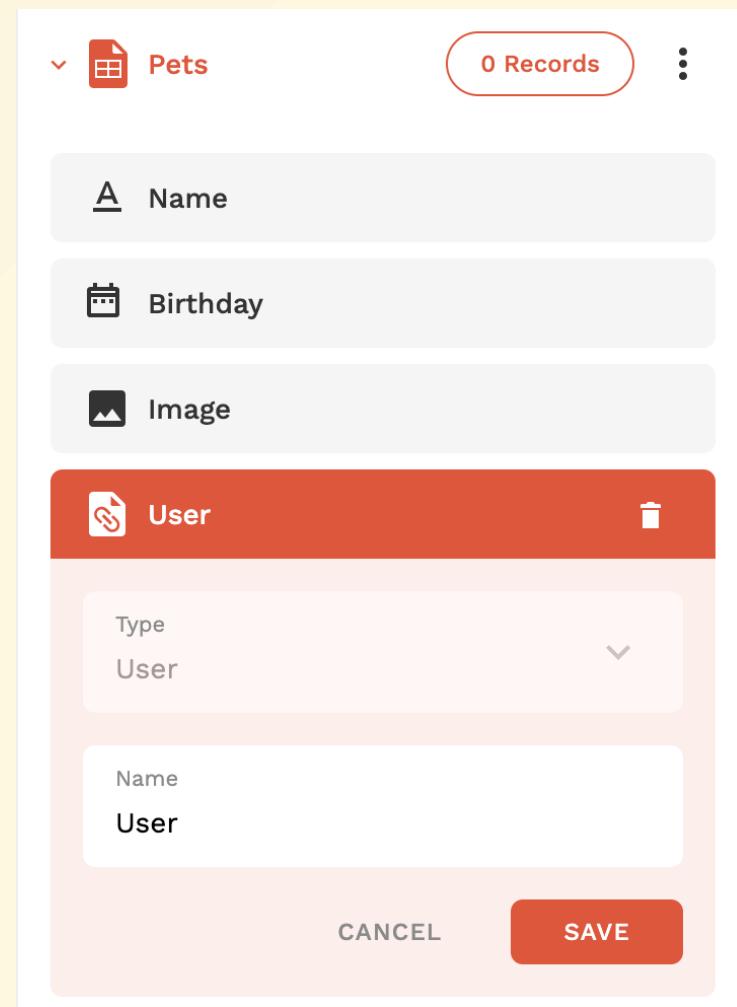
Pets trash icon

Type  
Pet

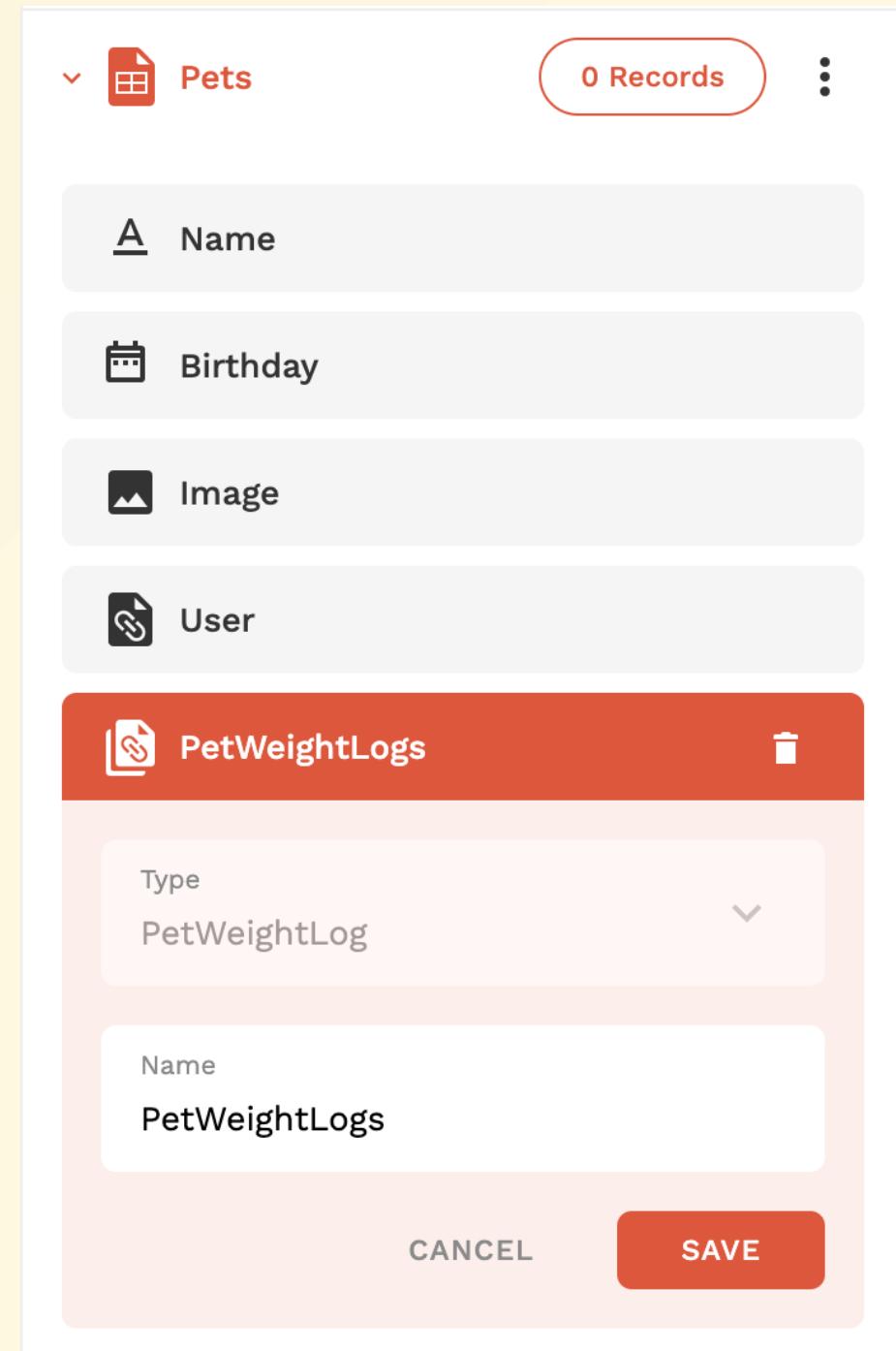
Name  
Pets

CANCEL SAVE

- Pets Collectionを確認すると、Users Collection側でRelationshipの設定をしたので、自動でUsers CollectionとのRelationshipが追加されています。
  - Users Collection側が1なので、末尾のsが省略されて、UserというProperty名になっています。

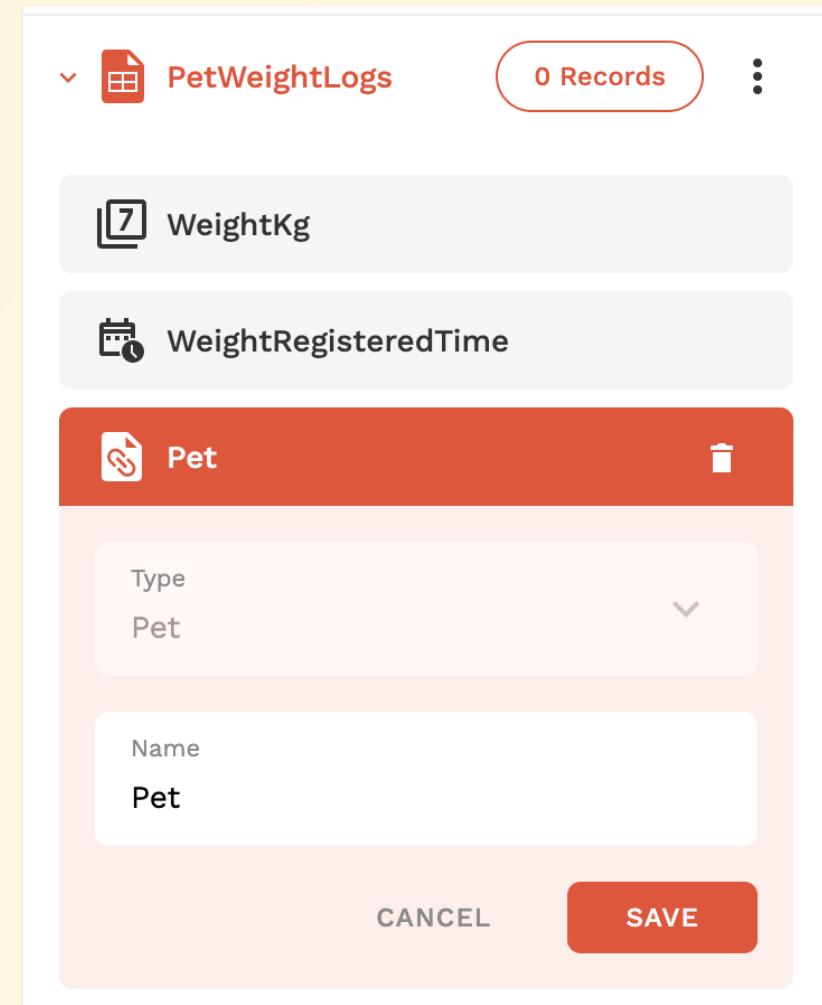


- Pets Collectionに、PetWeightLogs CollectionとのRelationshipを追加しましょう。
  - Pets Collectionを選択して、PetWeightLogs Collectionとの1対多のRelationshipを追加します。



PetWeightLogs Collectionを確認すると、Pets Collection側でRelationshipの設定をしたので、自動でPets CollectionとのRelationshipが追加されています。

- Pets Collection側が1なので、末尾のsが省略されて、PetというProperty名になっています。



以上がデータベース設計の流れです。

参考: Pets Collectionにペットの体重とその登録日時を含めた場合どうなるか

以下のようにレコードが登録される。

Pets						
	A Name	Image	Birthday	7 Weight(kg)	WeightRegisteredTime	Created
<input type="checkbox"/>	みる		8/8/2018	5.2	November 3, 2021 12:00 AM	a few seconds ago
<input type="checkbox"/>	みる		8/8/2018	5.1	November 2, 2021 12:00 AM	a minute ago
<input type="checkbox"/>	みる		8/8/2018	5	November 1, 2021 12:00 AM	a minute ago

この場合、少し困ることが出てきます。

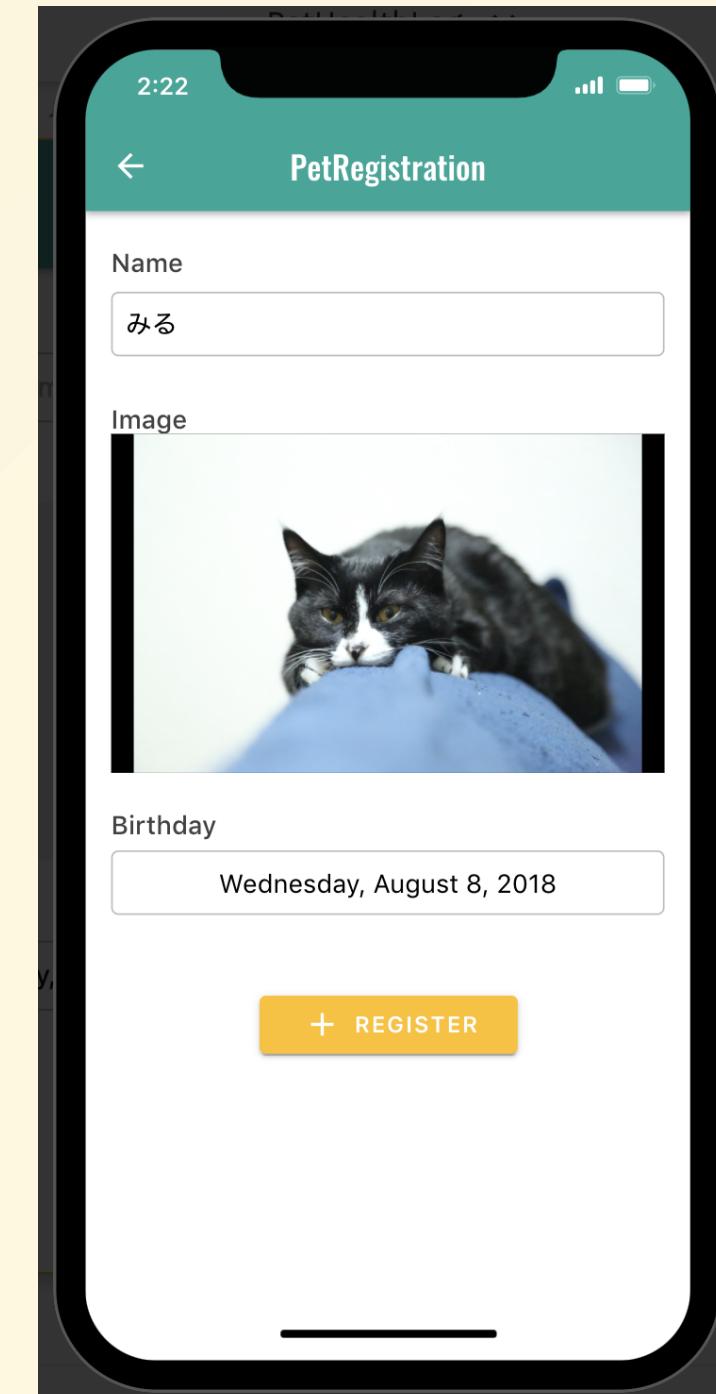
- 1匹のペットに対して異なるペットの体重とその登録日時が結合されたRecordが複数登録されるため、ペットの情報が重複して登録されてしまう。
  - 1匹のペットの名前を変更する時には、重複して登録されたそのペットについてのRecordを全て更新しないといけなくなり、処理が複雑になる。
- Adaloには一つのCollectionを選んでそこにRecordを登録するためのフォームを自動生成する便利な機能があるが、データを登録する単位でCollectionが分かれていないので、それが使えなくなる。

# データベース操作

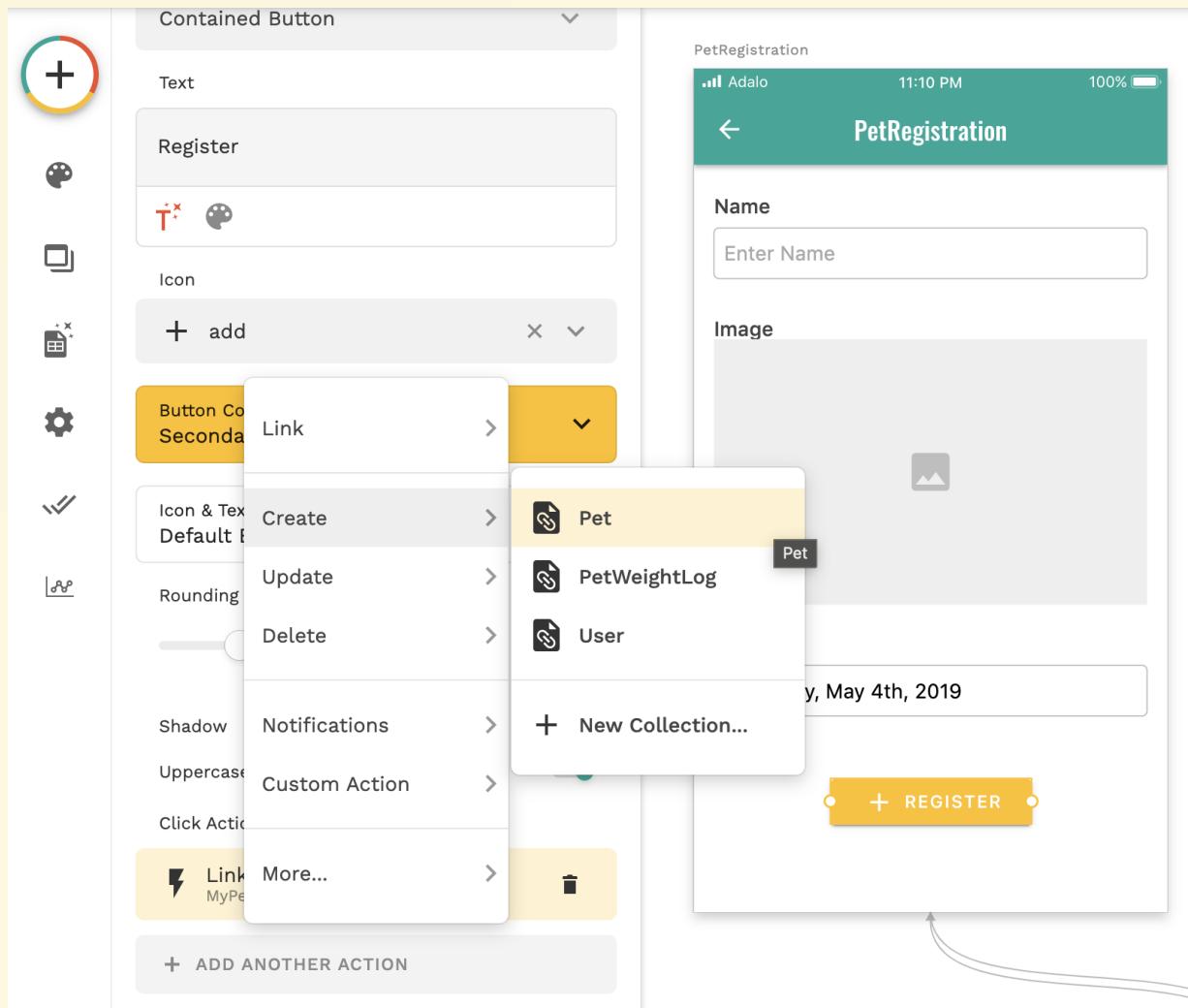
設計・構築したデータベースを使って、サンプルアプリでデータを操作できるようにしましょう。

## データの作成(CREATE)

まず、作成済みのペット登録画面で実際にペットのレコードを登録できるようにします。

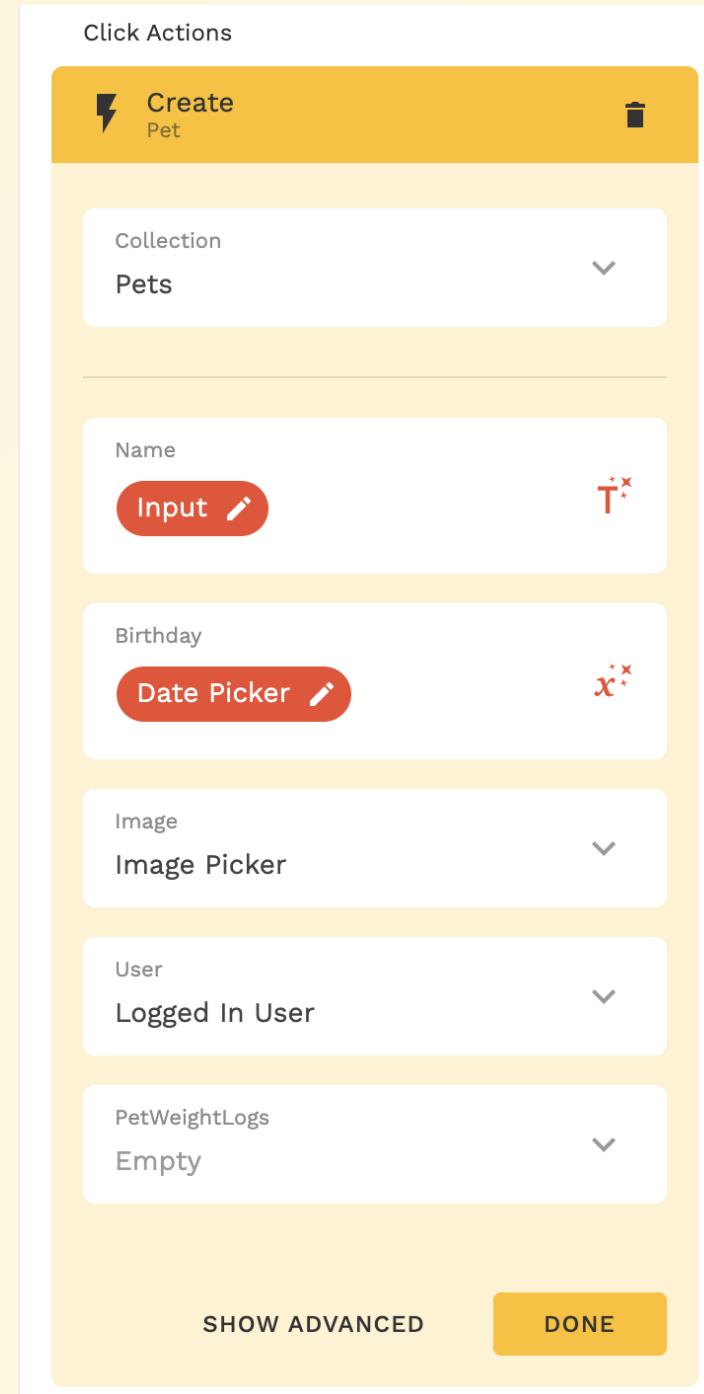


- ペット登録画面のREGISTERボタンを選択し、「ADD ANOTHER ACTION」をクリック
- Create > Pet を選択



以下を入力してDONE。

- NameはOther ComponentsのInputを選択
- BirthdayはOther ComponentsのDate Pickerを選択
- ImageはOther ComponentsのImage Pickerを選択
- UserはLogged In Userを選択
- PetWeightLogsはEmptyのまま(ペット登録時には不要)



Preview機能でペットを登録してみましょう。  
Pets CollectionにRecordが登録されたらOKです。

Pets

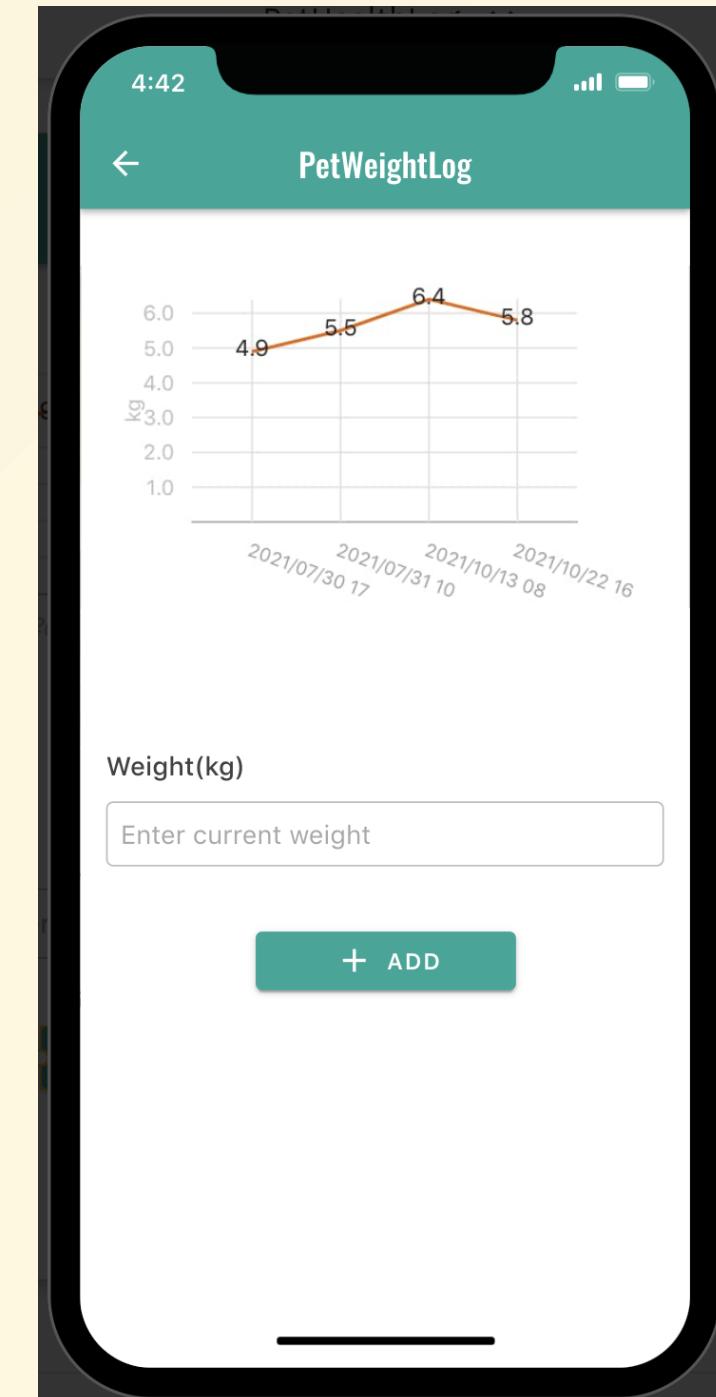
+ ADD PET

↑ ↓ <> 🔎

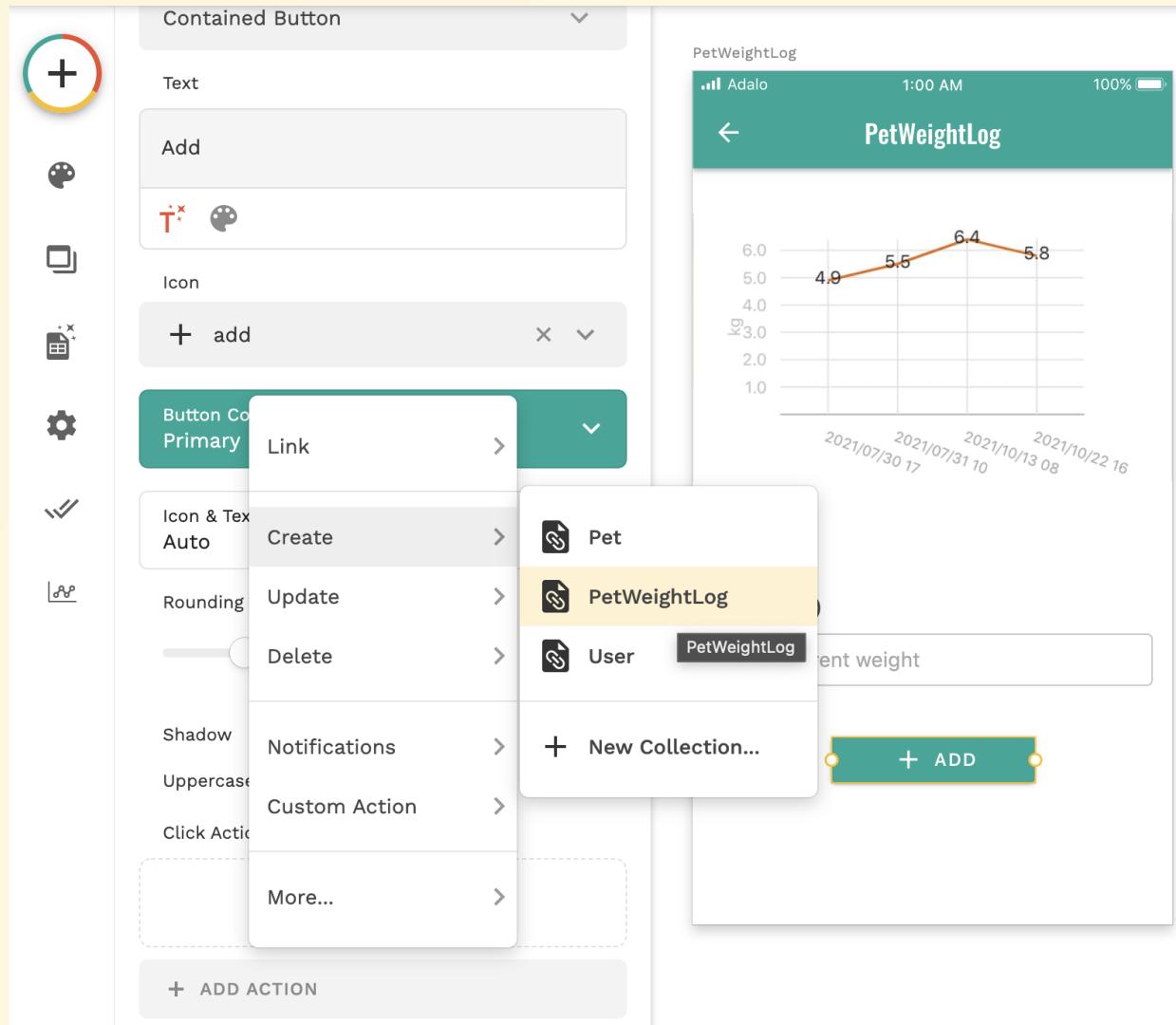
<input type="checkbox"/>	Name	Birthday	Image	User	PetWeightLogs	Created
<input type="checkbox"/>	みる(Database Record)	8/8/2018		imahashi@guildworks.jp		43 minutes ago

DONE

次に、ペットの体重管理画面で現在の体重を登録できるようになります。

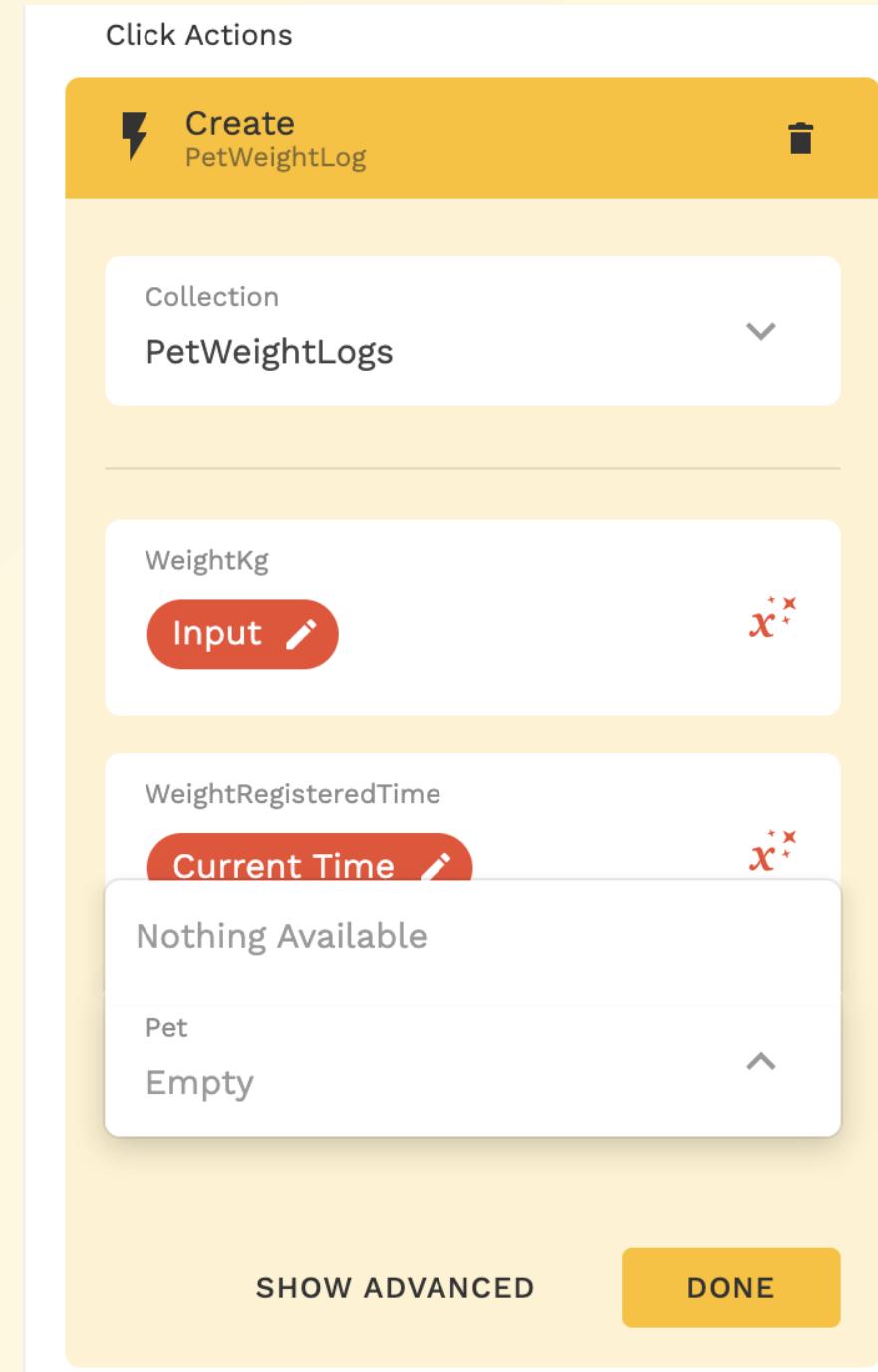


- ペットの体重管理画面で ADDボタンを選択し、「ADD ACTION」をクリック
- Create > PetWeightLog を選択



以下を入力してDONE。

- WeightKgはOther ComponentsのInputを選択
- WeightRegisteredTimeは Date&Time > Current Time を選択
- PetはNothing Availableなので、一旦Emptyのままにする  
(後ほど、選択したペットの体重を登録できるように設定します)

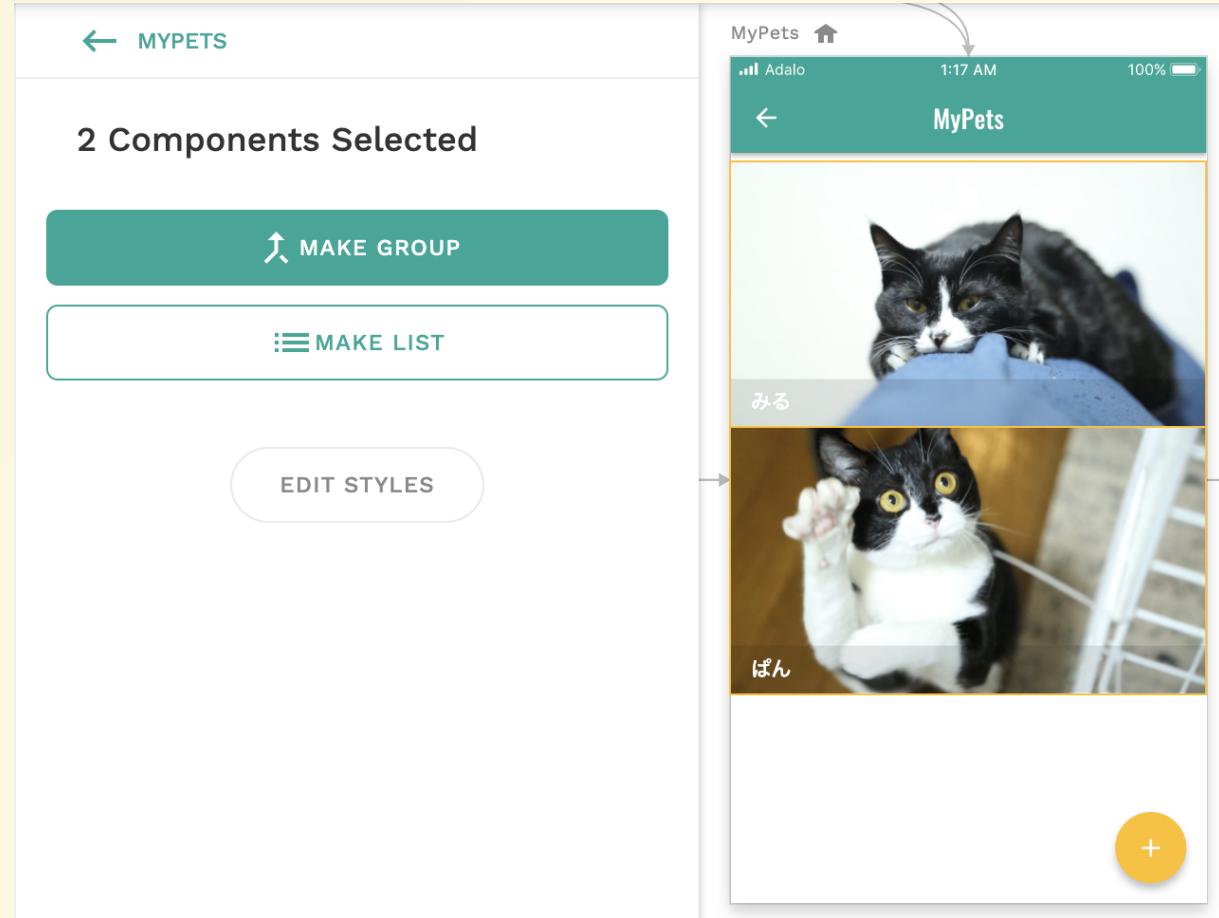


## データの表示(READ)

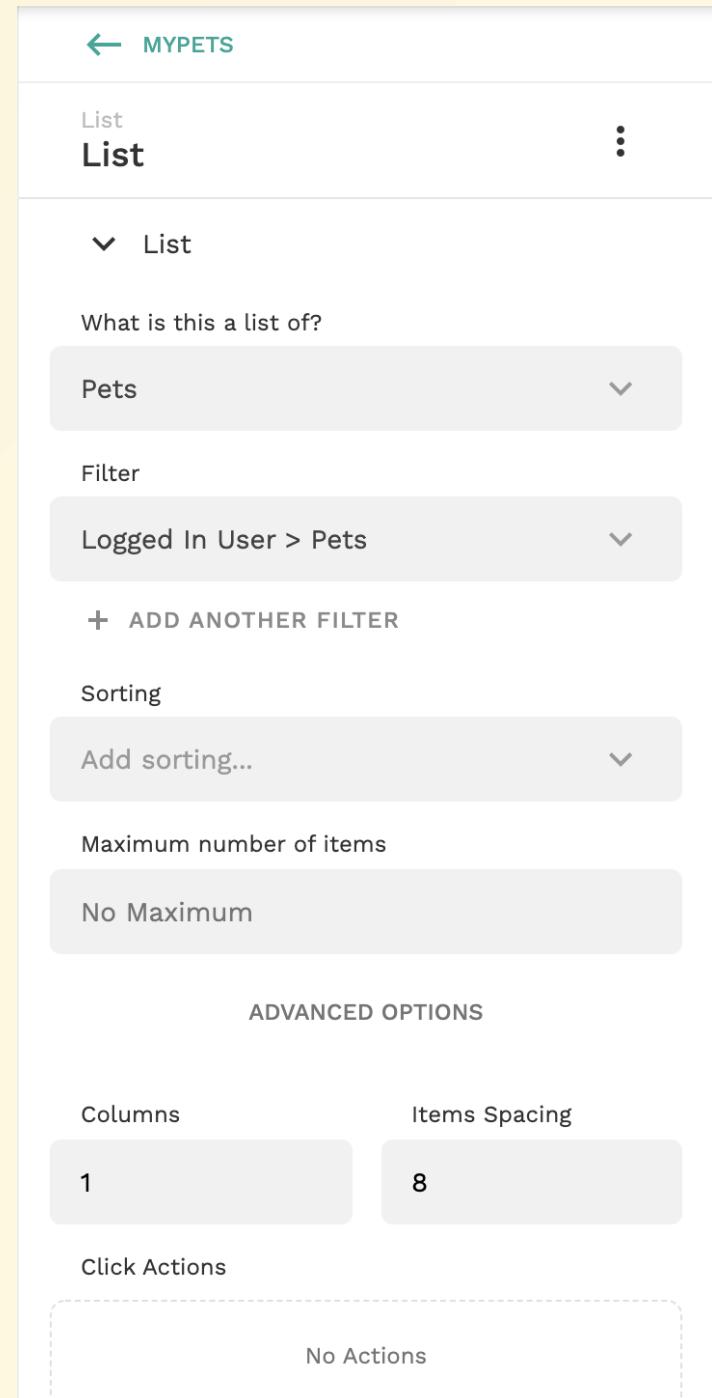
まず、作成済みのペット一覧画面に実際に登録したペットが表示されるようにします。



- ペットの画像と名前を表示している2つのコンポーネントを選択し、MAKE LISTをクリック



- What is this a list of?で Pets を選択
- Filterで Logged In User > Petsを選択

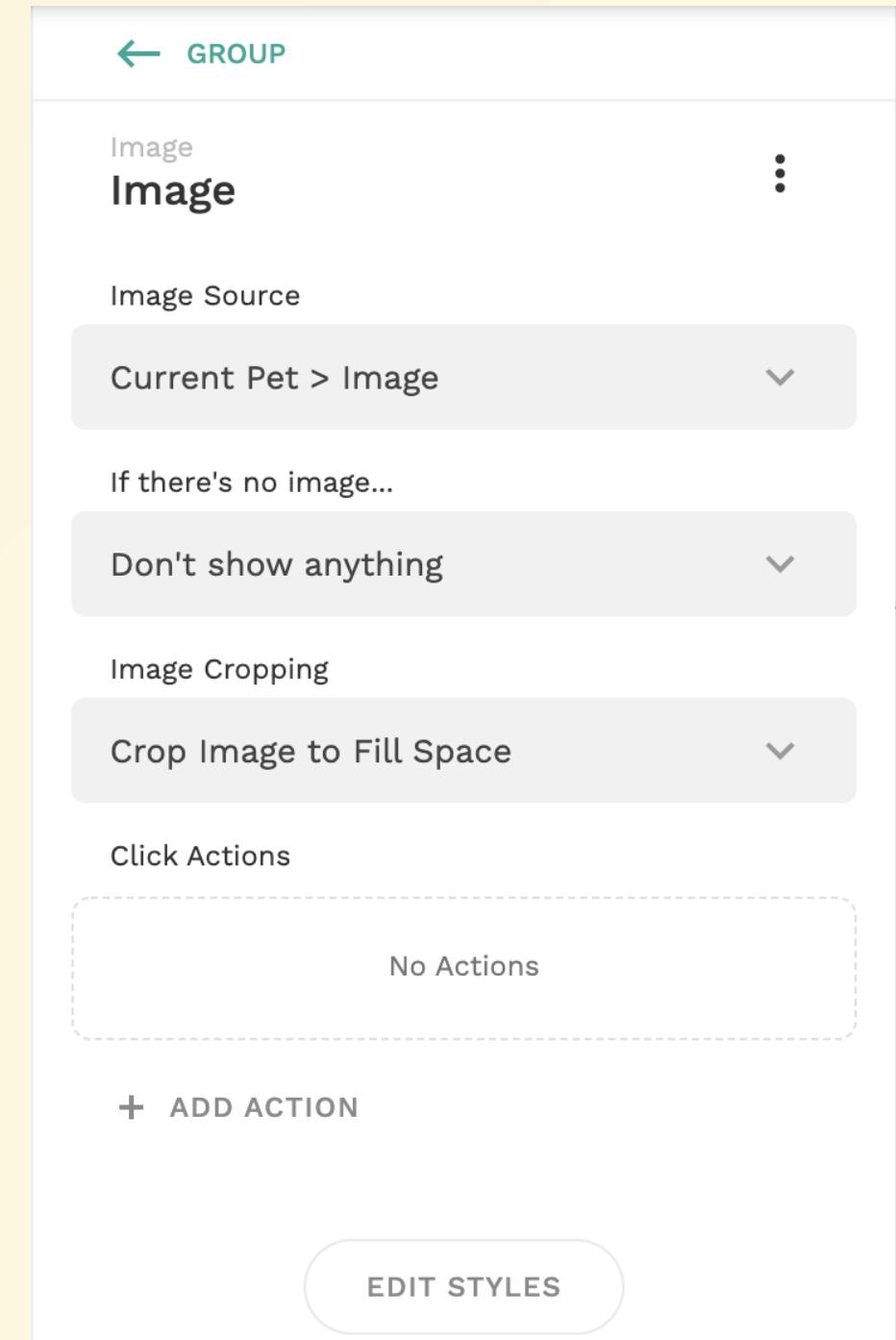


- 作成したListを構成するコンポーネントであるGroupの1つ目をクリック。
- Group内のImageコンポーネント、ペット名のコンポーネントをそれぞれ編集していきます。

The screenshot shows two panels from the Figma interface. On the left, the 'Components' panel for the 'List' component is displayed. It lists two items: 'Group' (highlighted with a gray background) and 'Group 3'. Below the list is a button labeled '+ ADD COMPONENT'. On the right, the detailed view of the 'Group' component is shown. It contains three items: 'Image' (with a preview icon), 'Shape' (set to 'Rectangle'), and 'Text' (containing the Japanese character 'みる'). Each item has a right-pointing arrow indicating it can be edited.

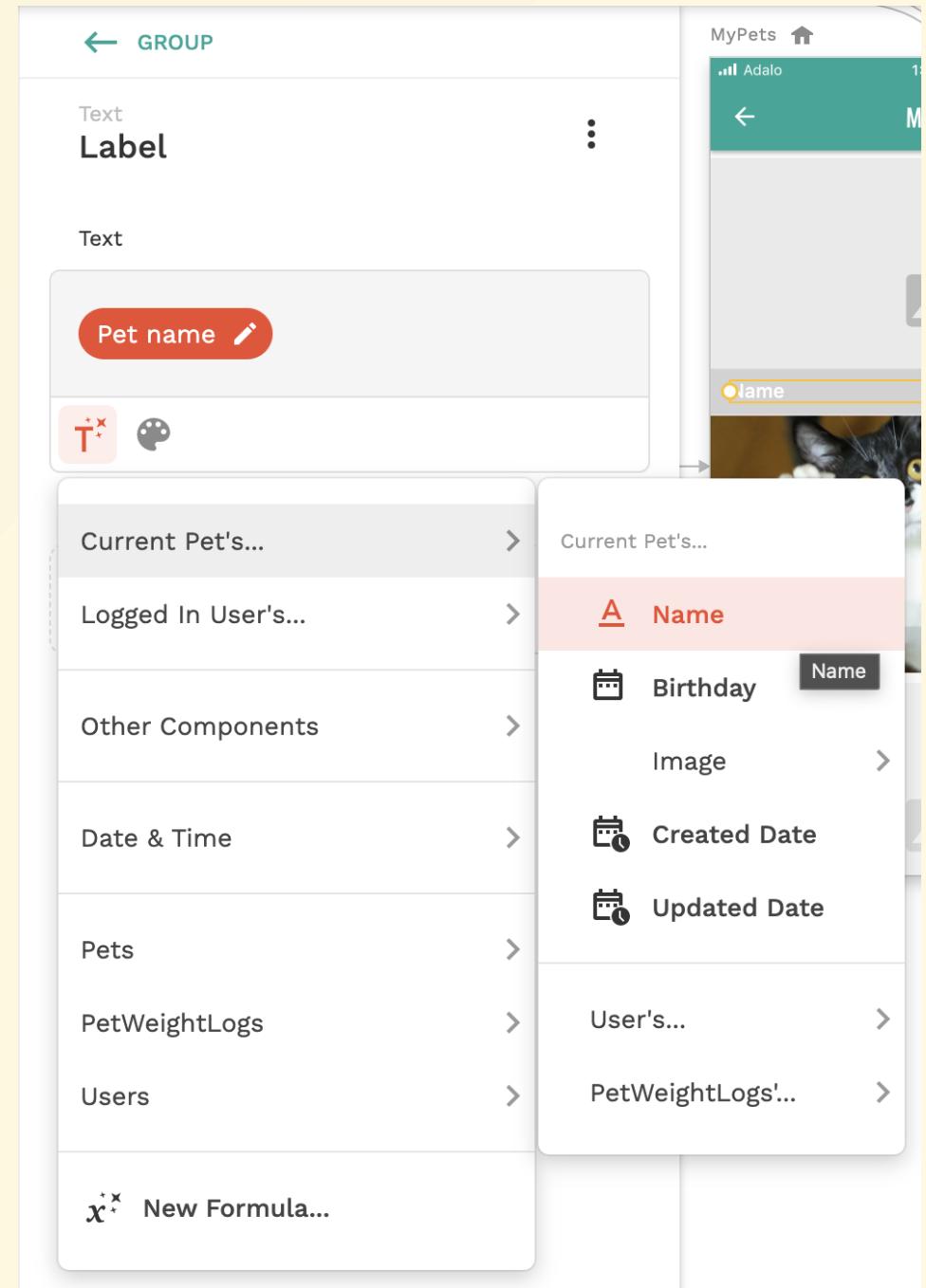
# Imageコンポーネントを編集

- Image Sourceで、Database > Current Pet's > Imageを選択
- If there's no image...でDon't show anythingを選択
  - あるいは、Show a place holder imageを選択して好きなペットのシルエット画像を設定してもOKです

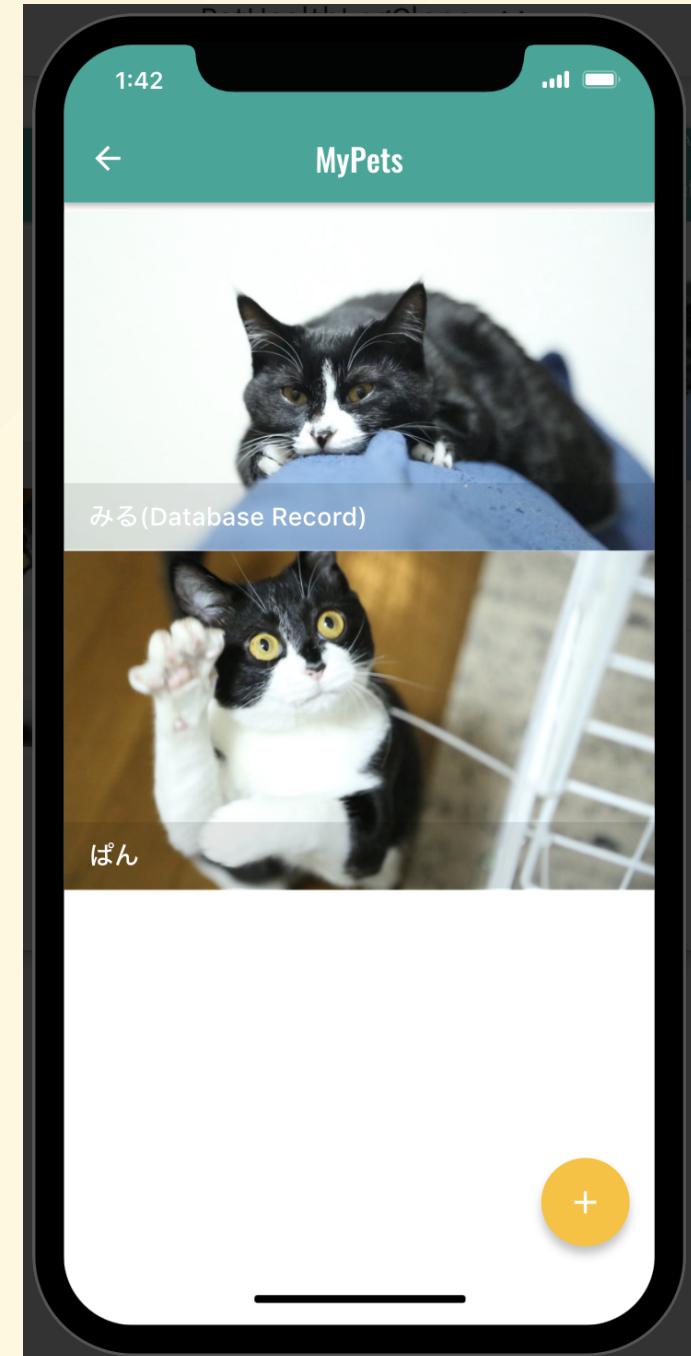


# ペット名のコンポーネントを編集

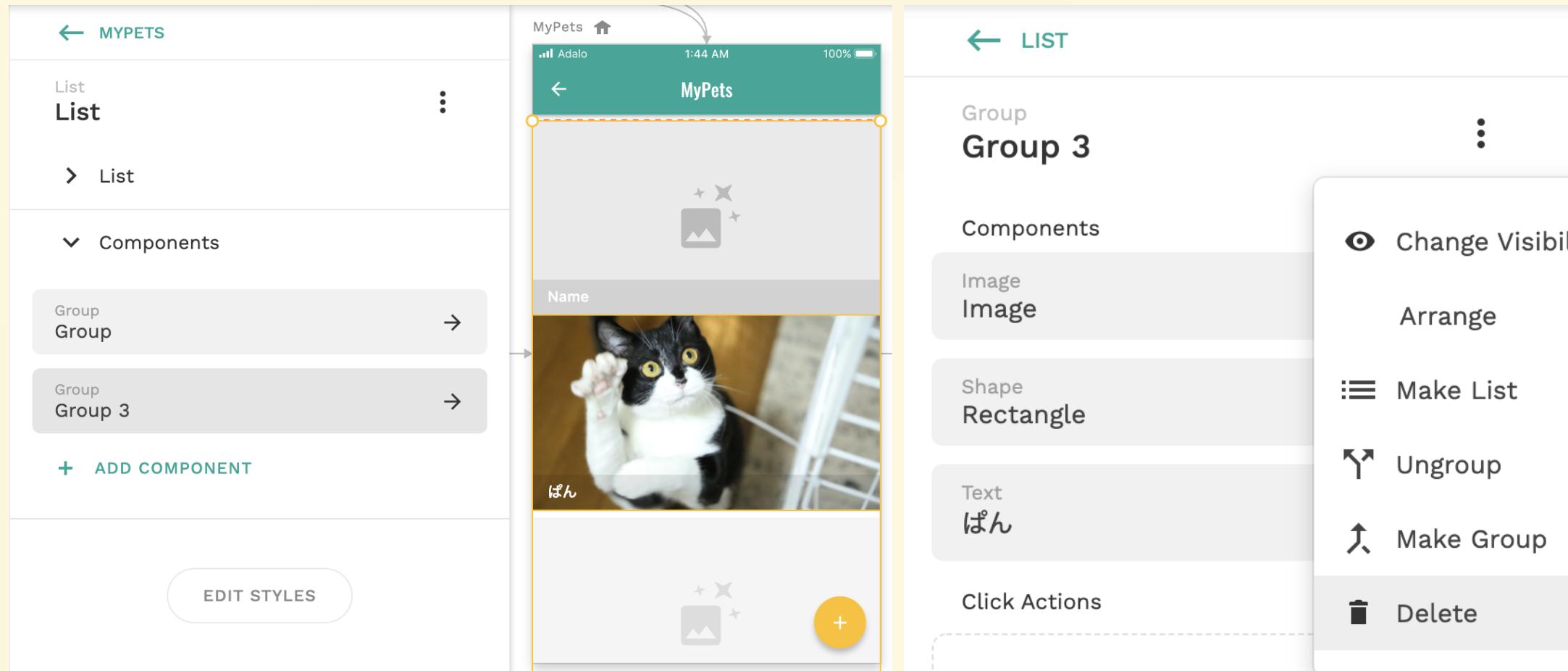
- Add Magic TextでCurrent Pet's > Nameを選択



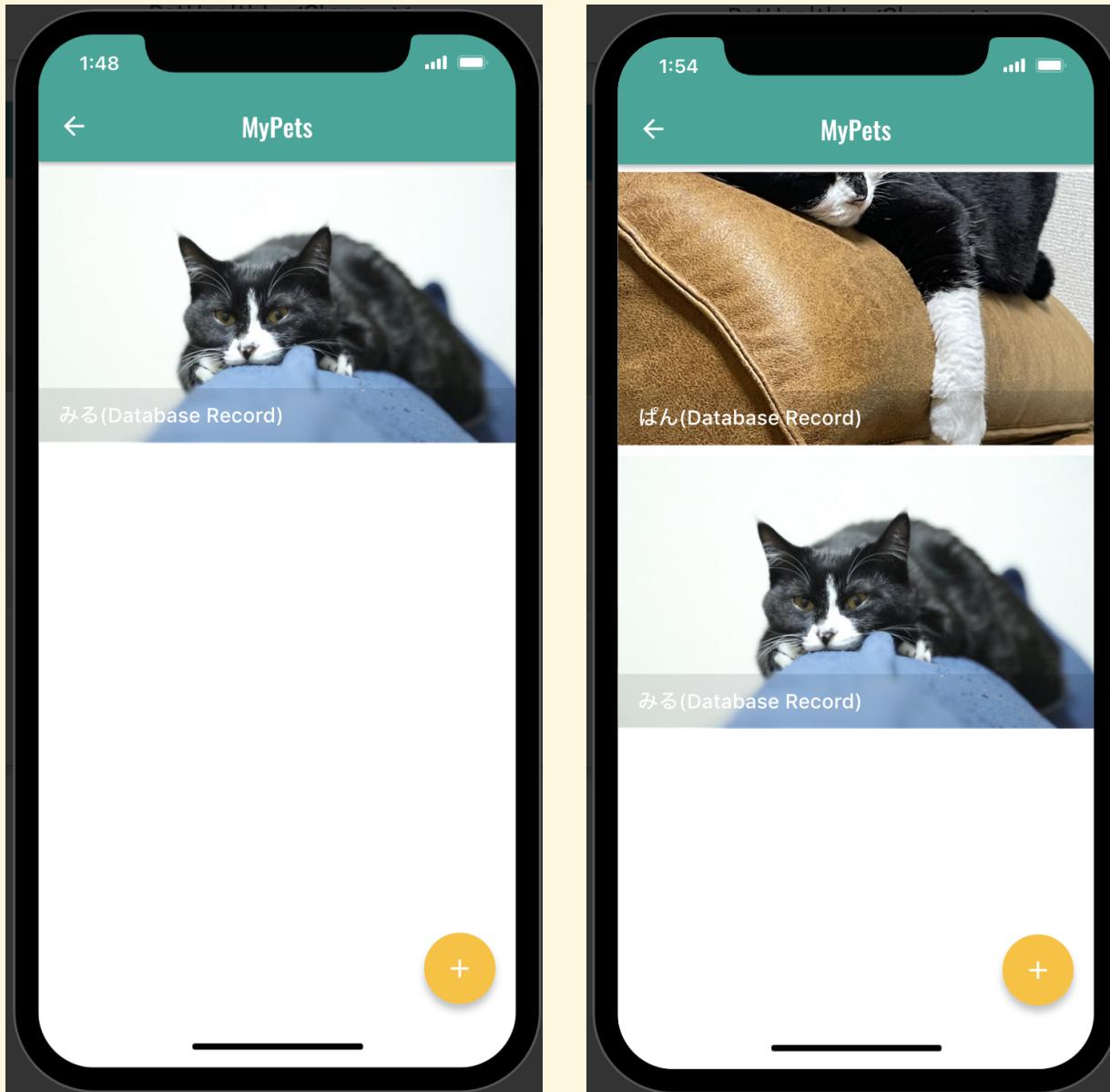
- Preview機能で確認すると、1匹目のペットとしてデータベースに登録したRecordが表示されます。



- ペットのListを構成するコンポーネントの中の2つ目のGroup(固定で表示していた2匹目のペット)は不要なので、削除しましょう

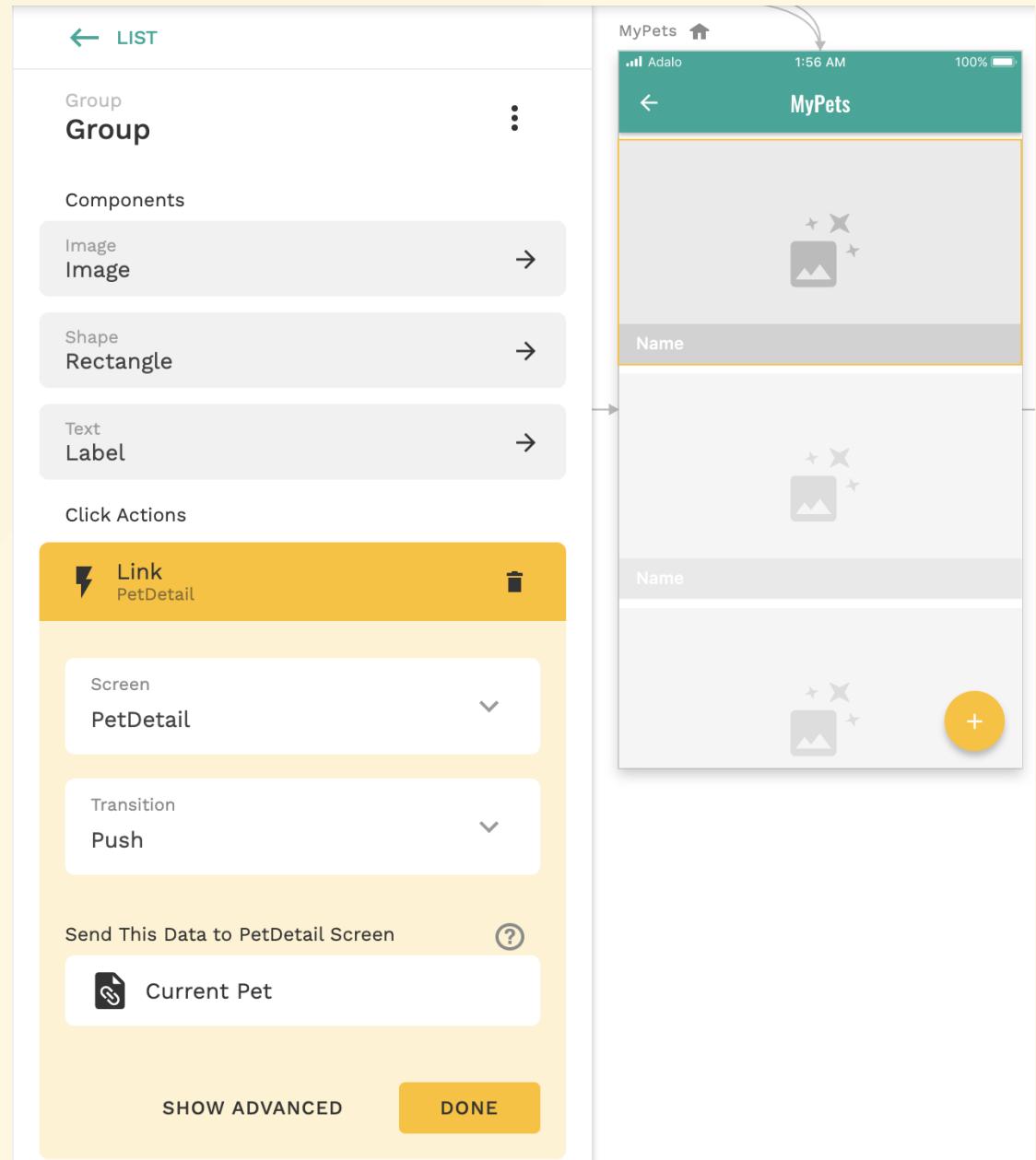


- Preview機能で確認すると、データベースに登録したペットだけが表示されるようになりました。
  - ペットを追加で登録すれば、複数のペットが表示されます

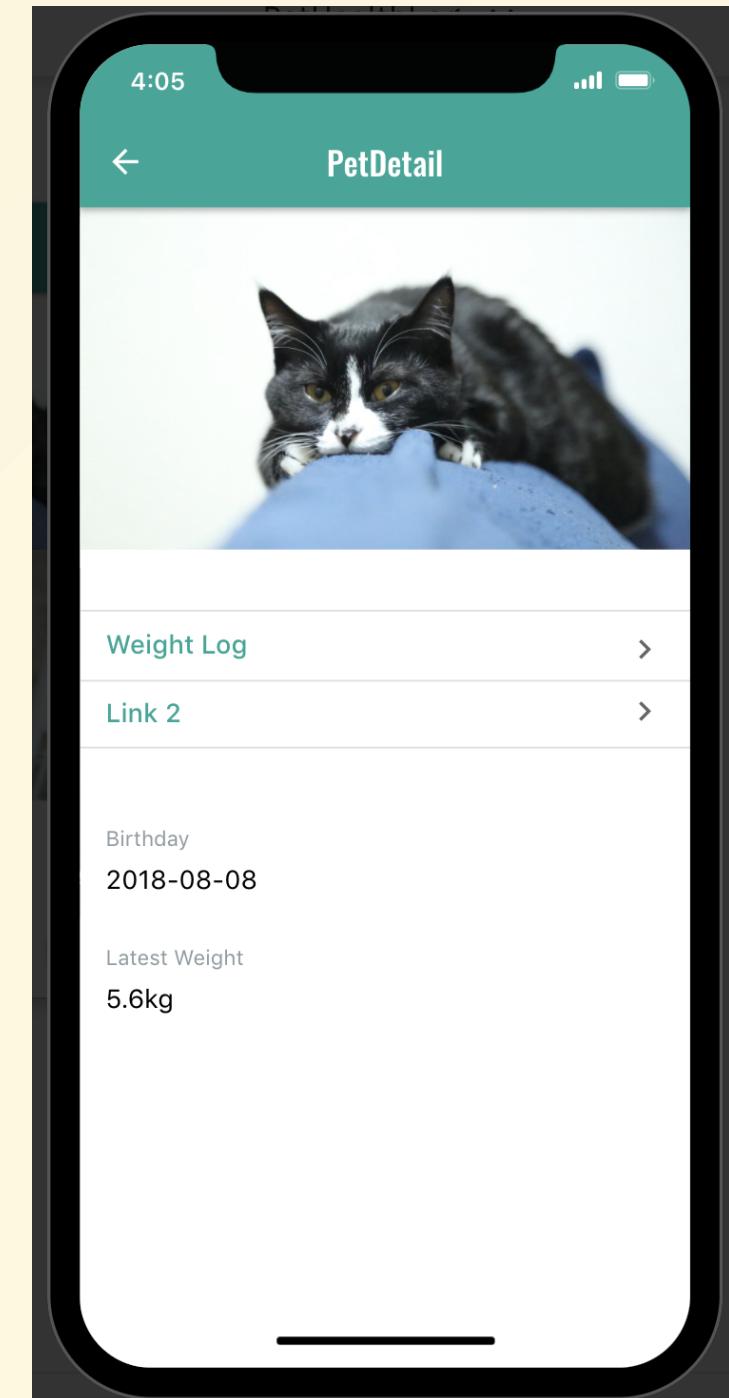


ペット一覧でペットをクリックした時に、そのペットの詳細画面に遷移できるようになっていることを確認します。

- ペットのGroupコンポーネントに設定されているLinkのSend This Data to PetDetail ScreenにCurrent Petが自動で設定されています。

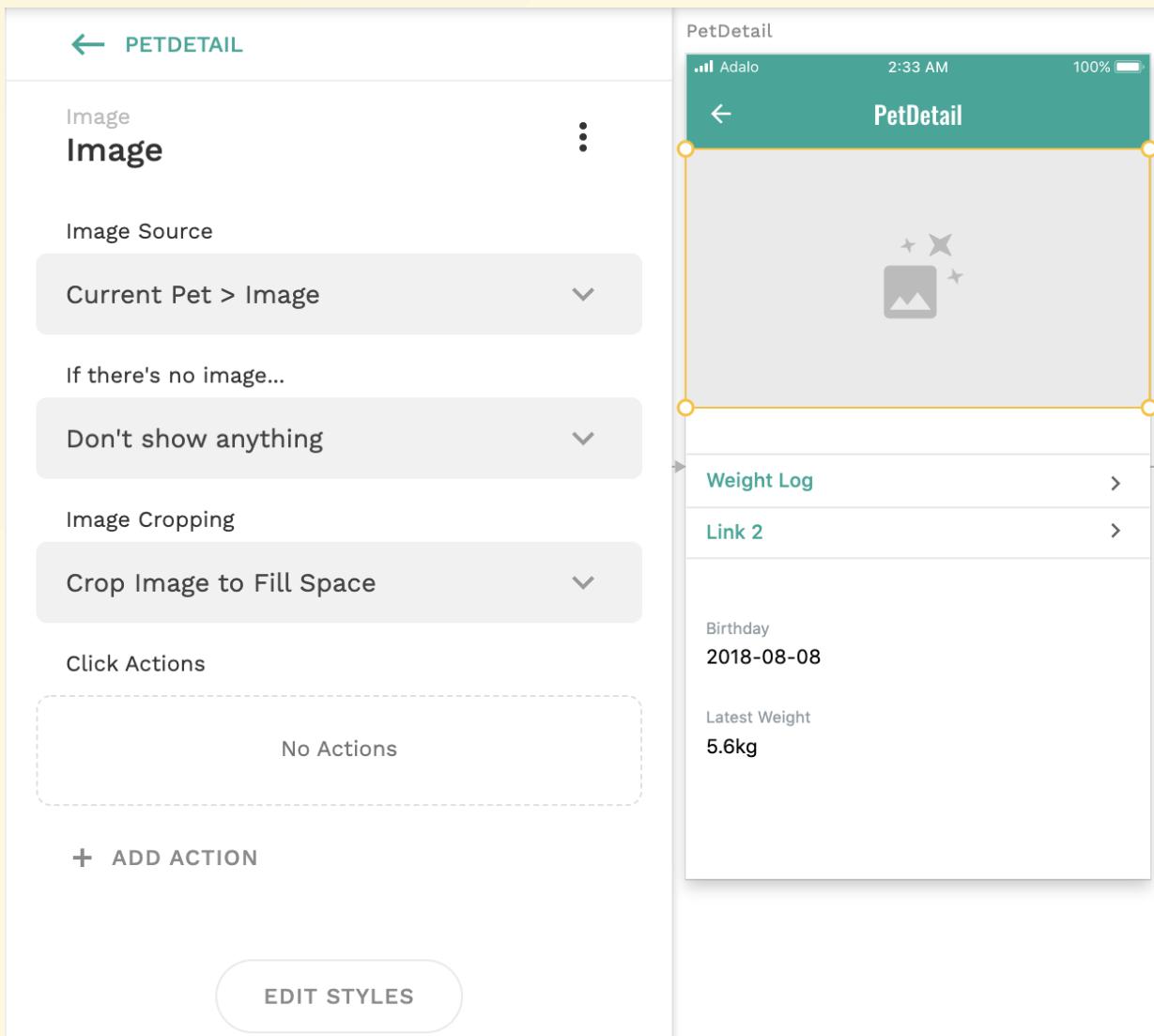


次に、ペット詳細画面にペット一覧画面で選択したペットが表示されるようにします。



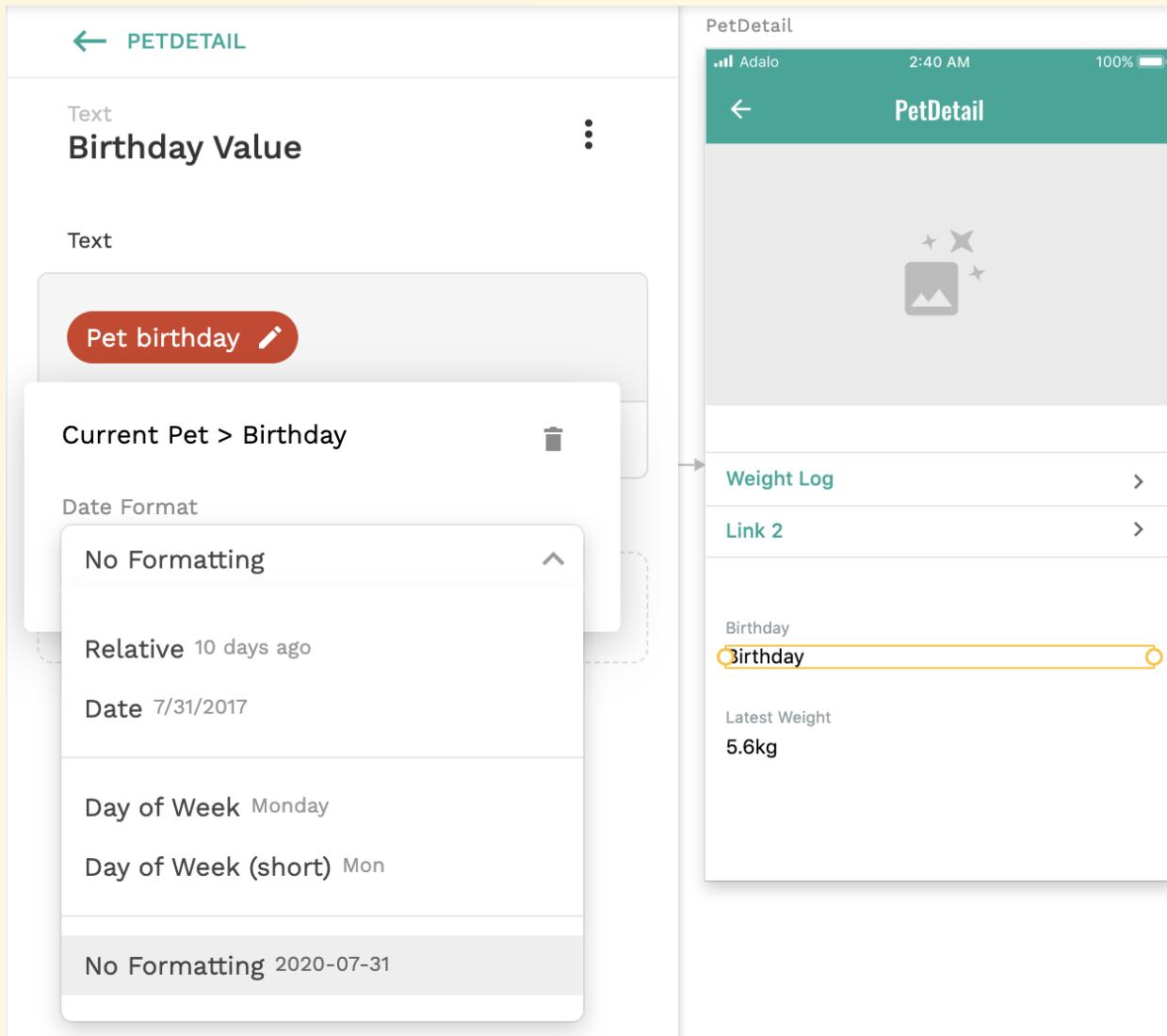
Imageコンポーネントをクリックし、

- Image SourceでDatabase > Current Pet's > Imageを選択
- If there's no image...でDon't show anythingを選択
  - あるいは、Show a place holder image を選択して好きなペットのシルエット画像を設定してもOKです



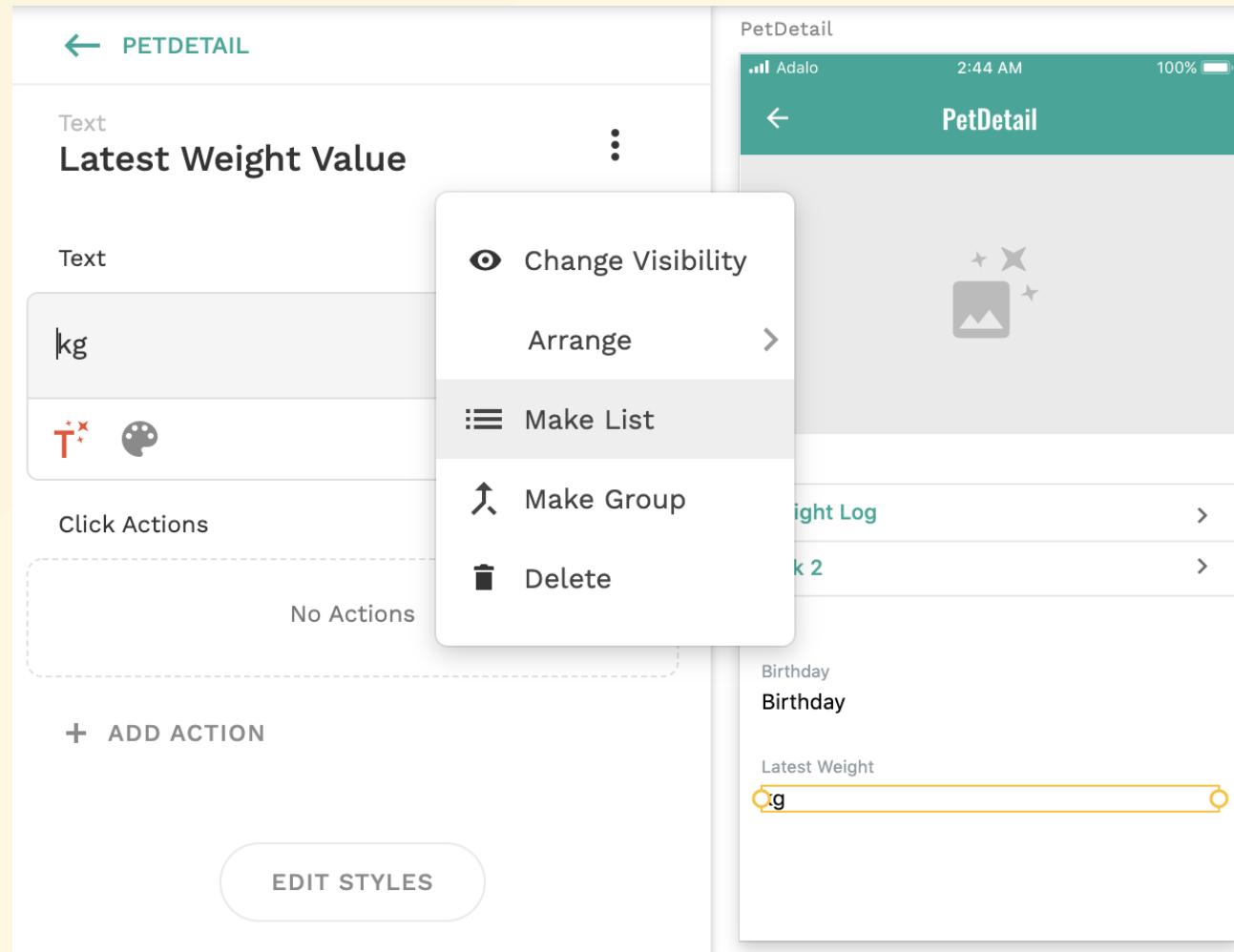
# Birthday Valueコンポーネント をクリックし、

- TextでCurrent Pet's > Birthdayを選択
- Date FormatでNo Formattingを選択



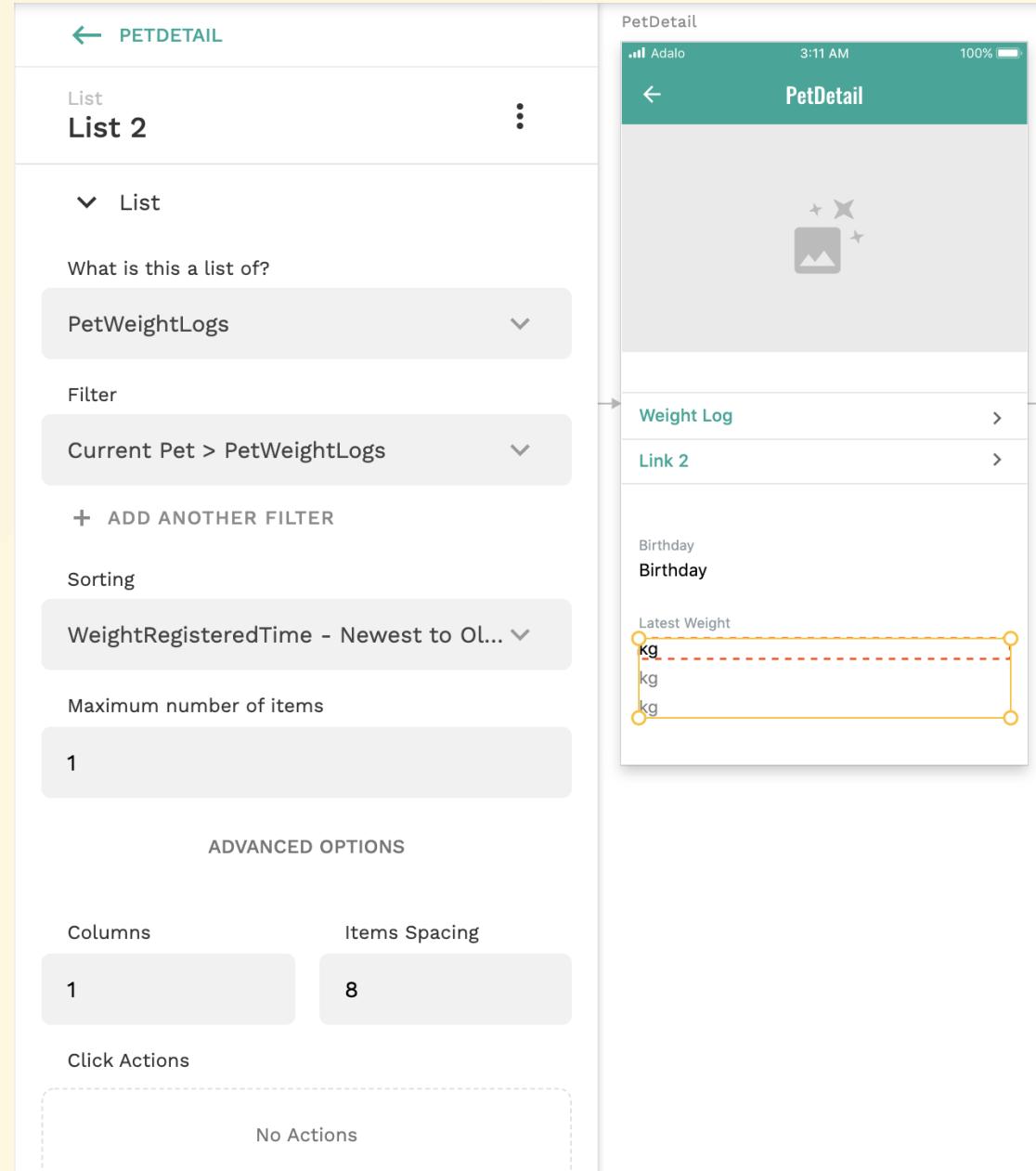
- Latest Weight Valueコンポーネントをクリックし、MAKE LISTでリストにする

\* 最新の1件を表示するためには、そのコンポーネントをListにします  
(次のページの設定で、最新の1件に絞り込みます)

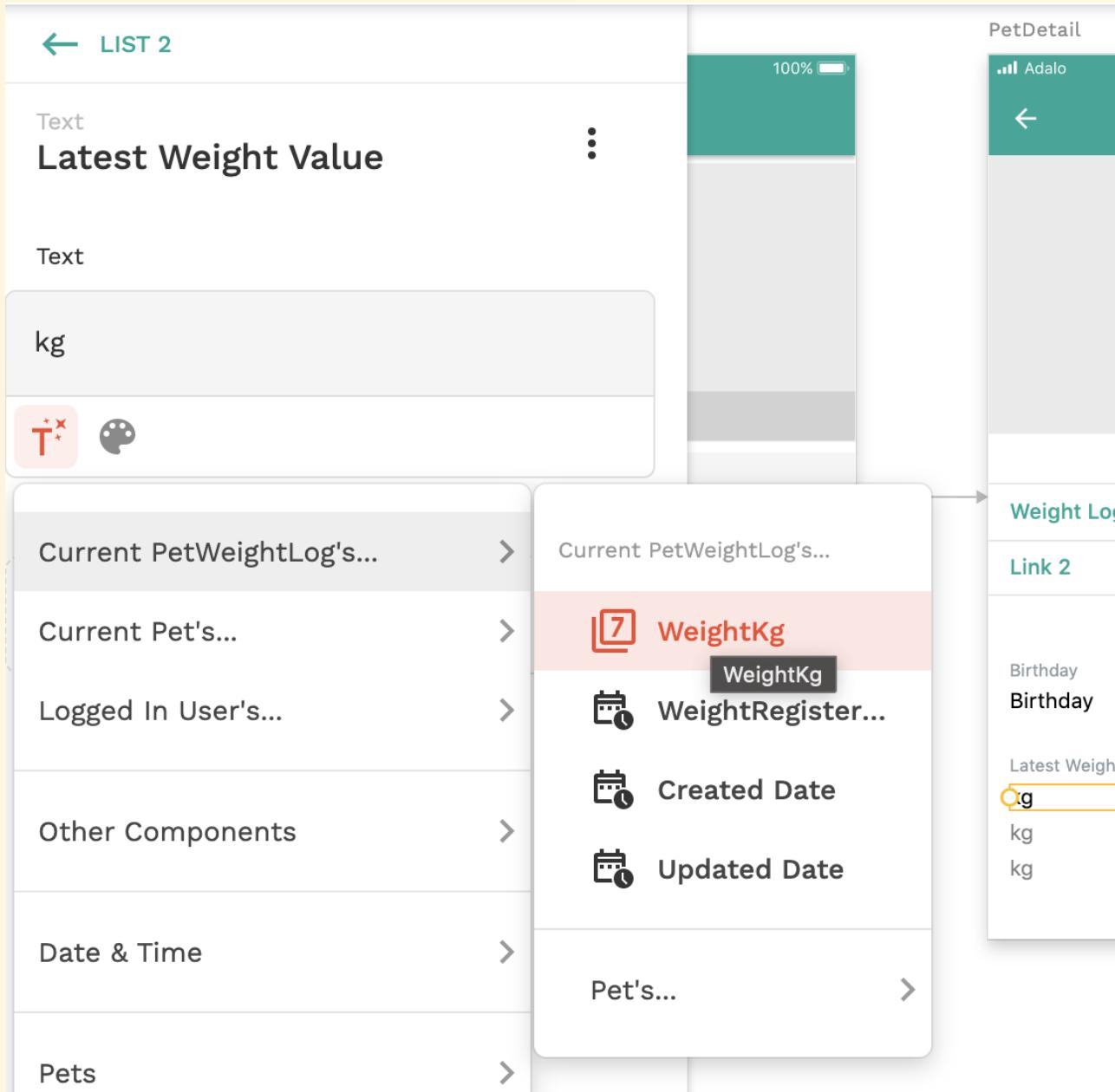


- What is this a list of?で PetWeightLogsを選択
- Filterで Current Pet > PetWeightLogsを選択
- Sortingで WeightRegisteredTime - Newest to Oldestを選択
- Maximum number of itemsに1を設定

これにより、最新の1件だけに絞り込まれます。

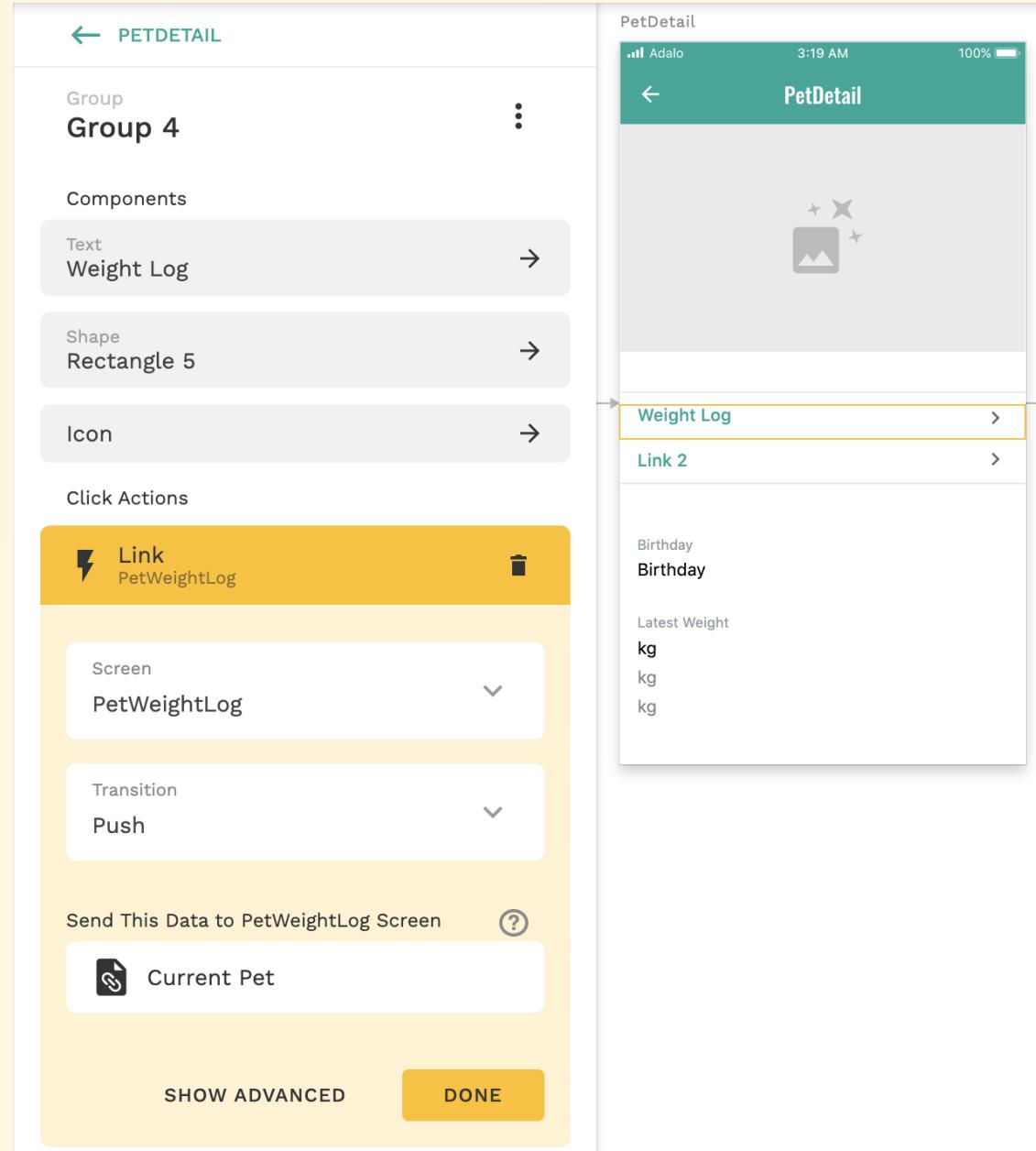


- Latest Weight Valueコンポーネントをクリックし、Textの"kg"の前にCurrent PetWeightLog's WeightKgを追加する



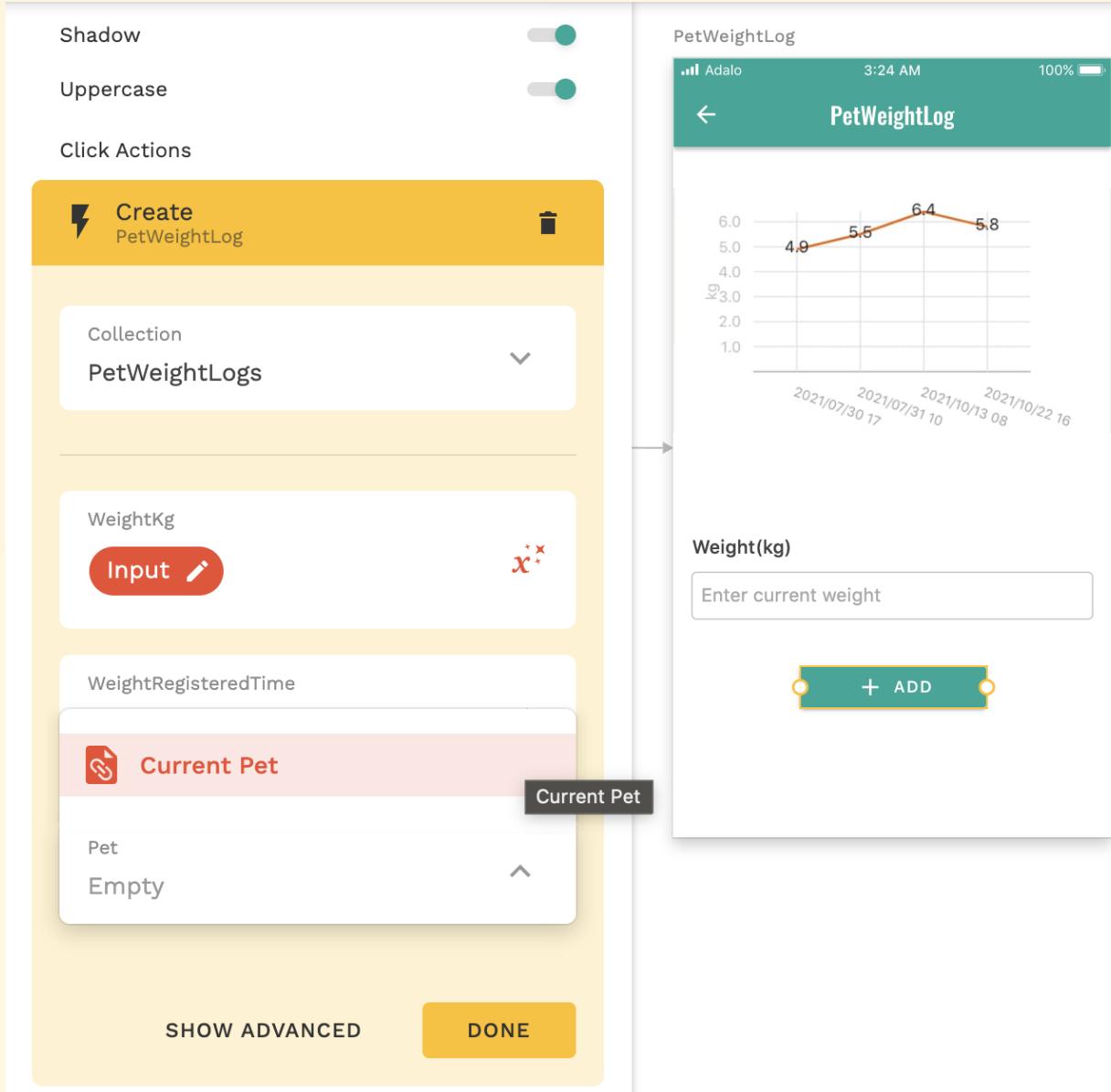
ペット詳細画面からペットの体重管理画面に遷移する際に選択したペットが受け渡されることを確認します。

- Weight Logというリンクのコンポーネントを選択し、LinkのSend This Data to PetDetail ScreenにCurrent Petが自動で設定されています。



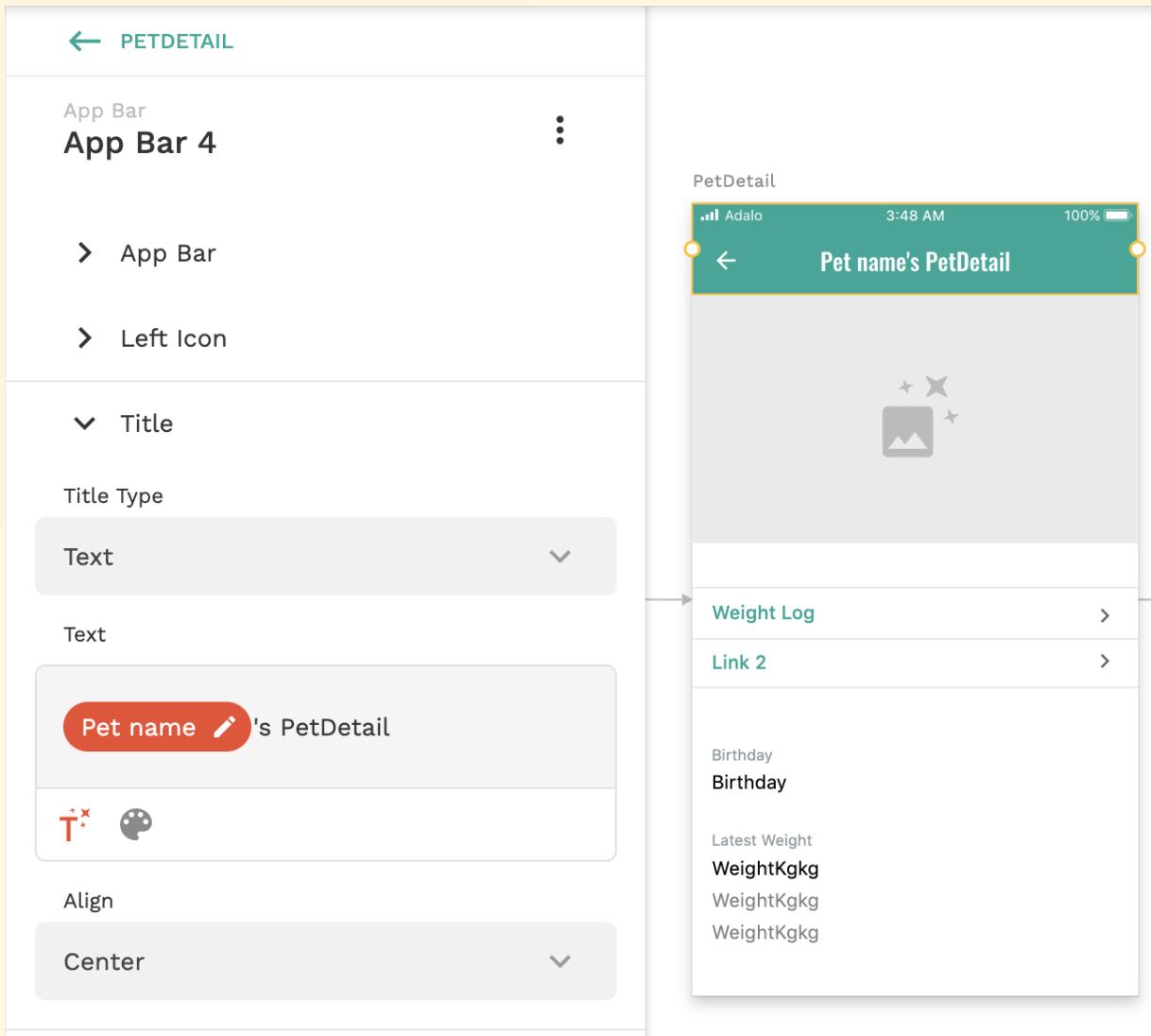
ペットの体重管理画面で選択したペットの体重をデータベースに登録できるようにします。

- ADDボタンを選択し、CreateのActionで一旦EmptyのままにしていたPetにCurrent Petを設定



ペット詳細画面、ペットの体重管理画面に選択したペットの名前を表示しましょう。それぞれの画面で、

- 画面上部のAppBarコンポーネントを選択し、TitleのTextに Current Pet's > Name を追加
- その後ろに、's という文字列を追加

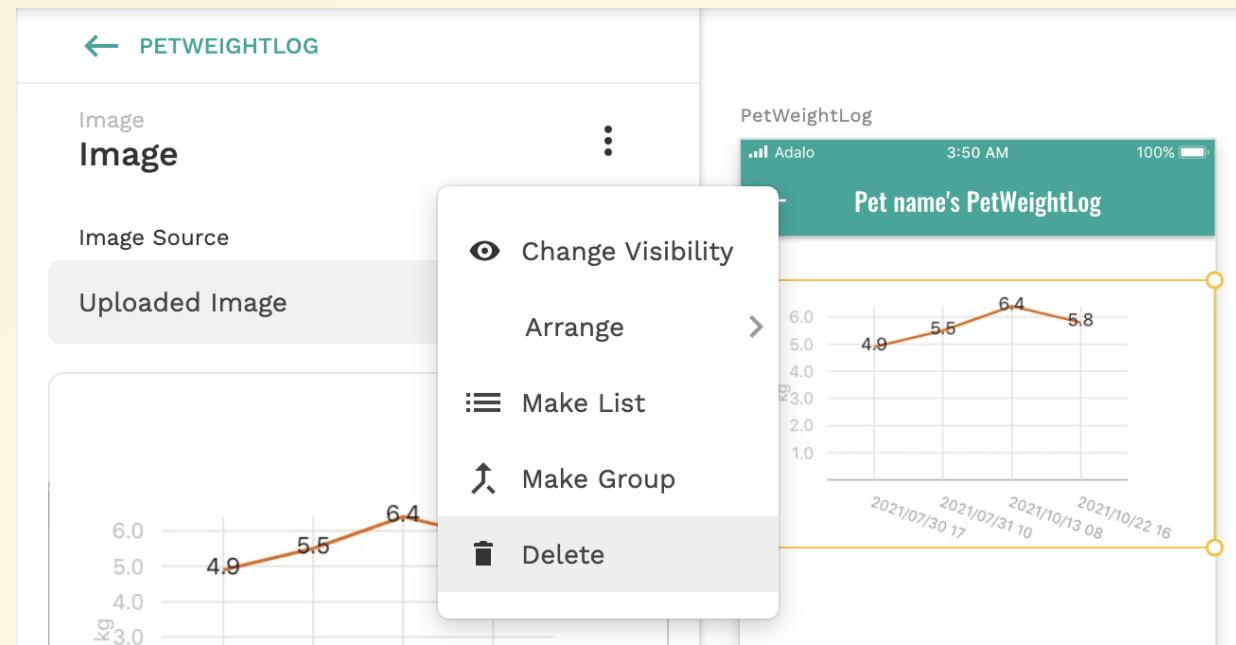


Preview機能で、以下を確認しましょう。

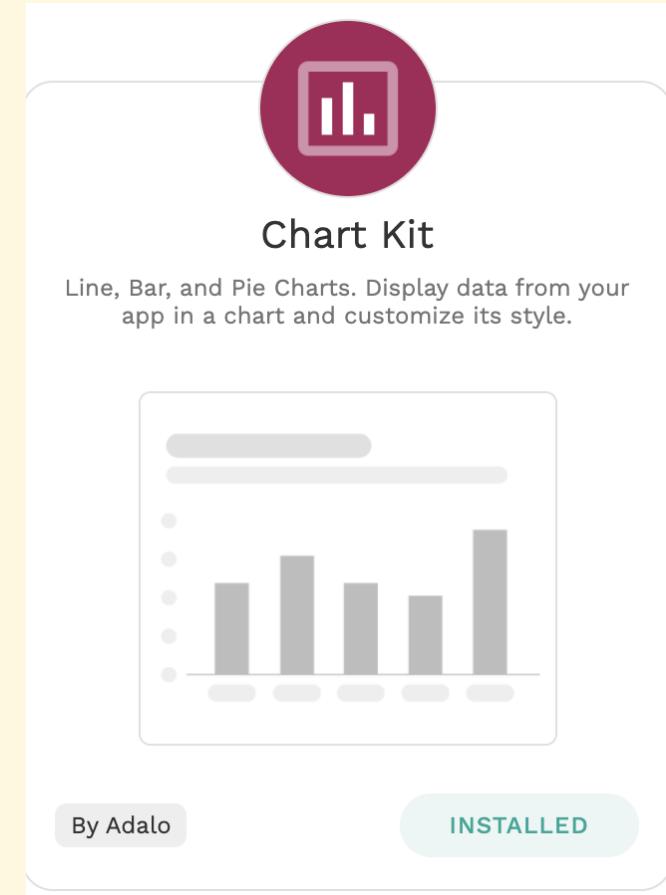
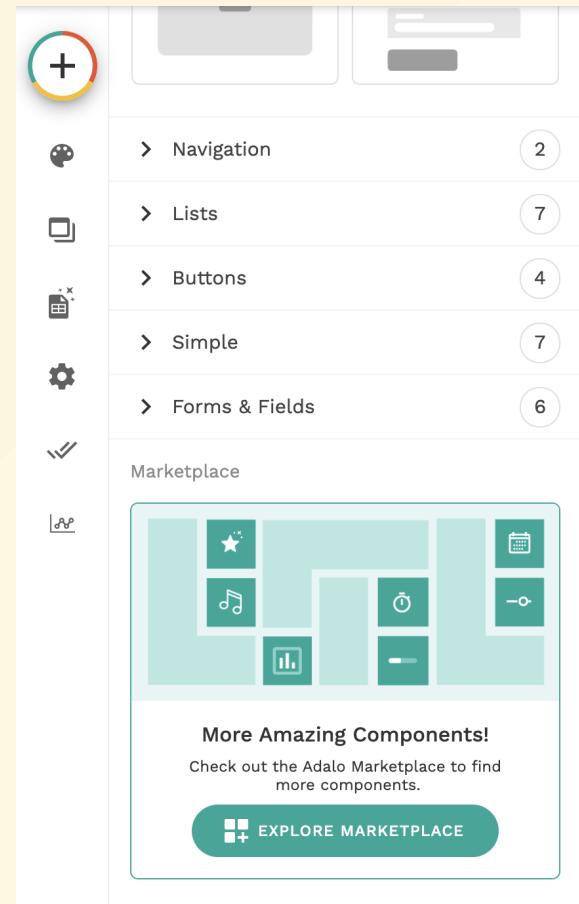
- ペット一覧画面から選択したペットの詳細画面に遷移できること
- ペット詳細画面から選択したペットの体重管理画面に遷移できること
- ペットの体重を登録すると、データベースのPetWeightLogs CollectionにRecordが登録されること
- ペット詳細画面に最新の体重が表示されること

次に、ペットの体重管理画面で過去に登録したペットの体重がグラフとして表示されるようになります。

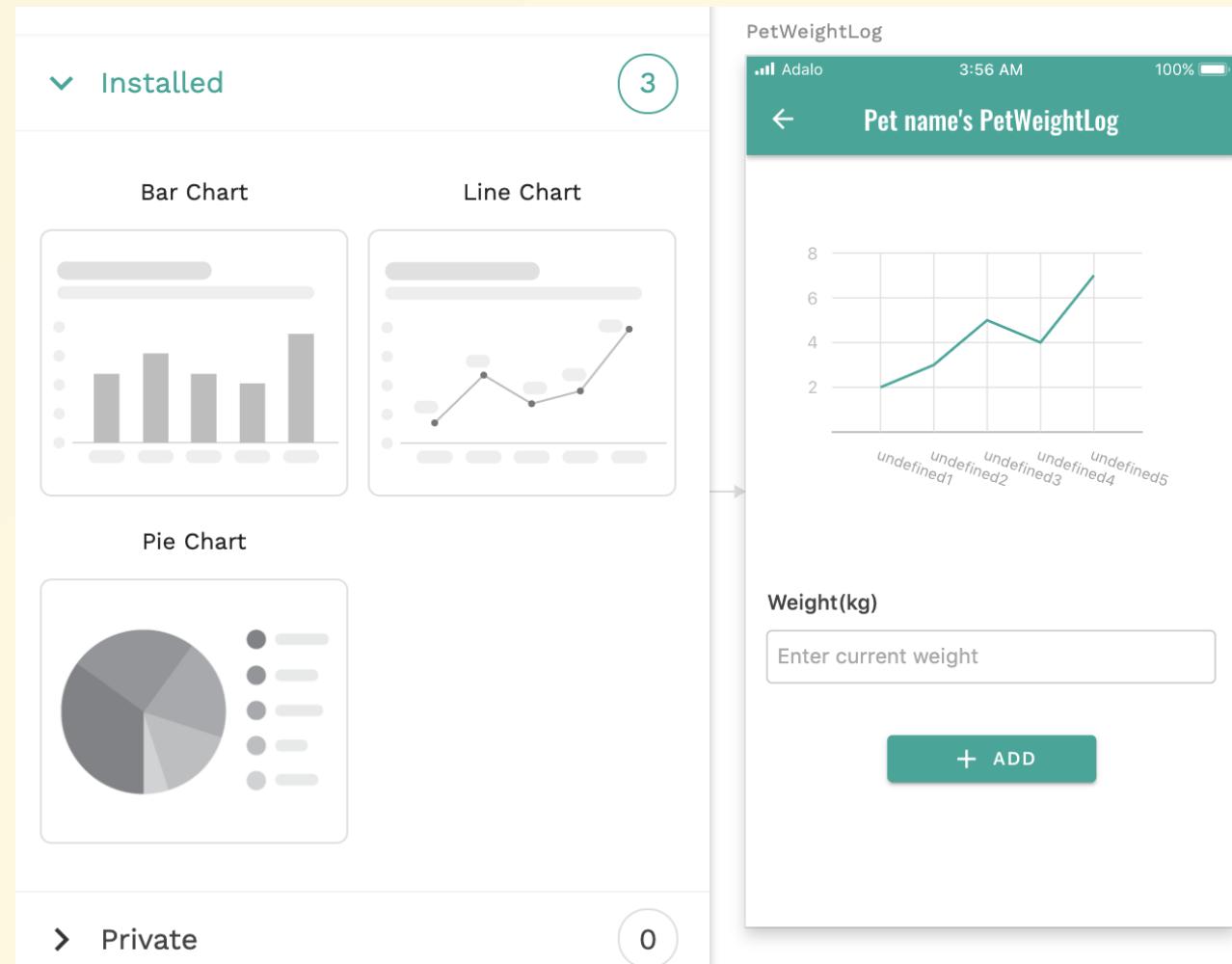
- まず、画像で貼り付けていたグラフを削除



- ADD COMPONENTから  
EXPLORE MARKETPLACE  
を選択
- Chart KitをINSTALL



- Line Chartを画面に追加



## Line Chartを設定します。

- What is this a chart of?で PetWEightLogsを選択
- FilterでCurrent Pet > PetWeightLogsを選択
- Custom Filterに WeightRegisteredTime Is after 30 days ago を設定
- Sortingで WeightRegisteredTime - Oldest to Newestを選択

Line Chart  
Line Chart

▼ Line Chart

What is this a chart of?

PetWeightLogs

Filter

Current Pet > PetWeightLogs

Custom Filter

WeightRegisteredTime

Is after

30 days ago

+ ADD ANOTHER FILTER

Sorting

WeightRegisteredTime - Oldest to Ne...

Maximum number of items

No Maximum

ADVANCED OPTIONS

- X Axis Valueに  
PetWeightLog >  
WeightRegisteredTimeを設  
定
  - Date FormatにDate /  
Timeを設定
- Y Axis Valueに  
PetWeightLog > WeightKg  
を設定

The screenshot displays a configuration interface for a visualization, likely a line chart. It consists of two main sections: 'X Axis Value' and 'Y Axis Value'.

**X Axis Value:**

- Selected value: PetWeightLog weightregisteredtime
- Text input field with a red 'x' icon for clearing.
- Y Axis Value section below it.

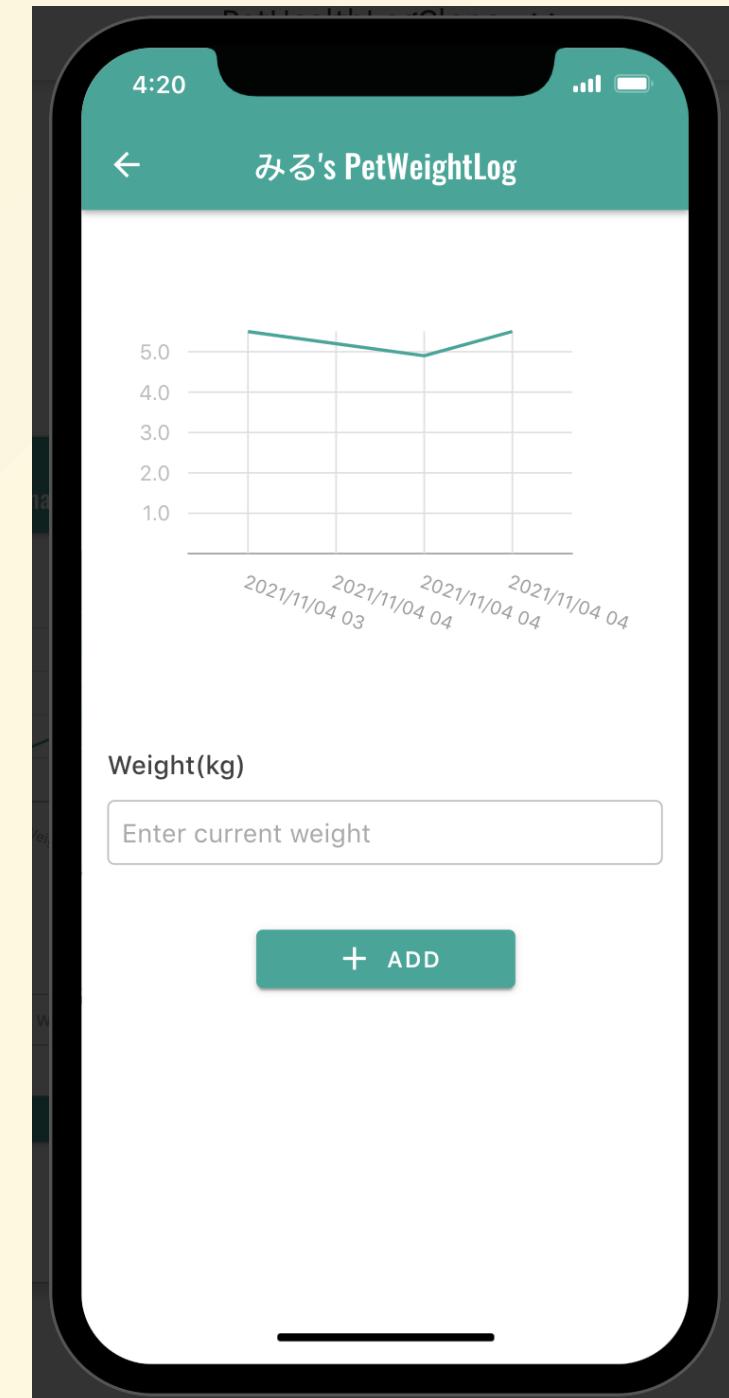
**Y Axis Value:**

- Selected value: PetWeightLog weightkg
- Text input field with a red 'x' icon for clearing.
- Point Action section:
  - No Actions
  - + ADD ACTION
- Styles section.

Preview機能でグラフの表示を確認しましょう。

- 体重を複数追加すると、グラフが描画されます。

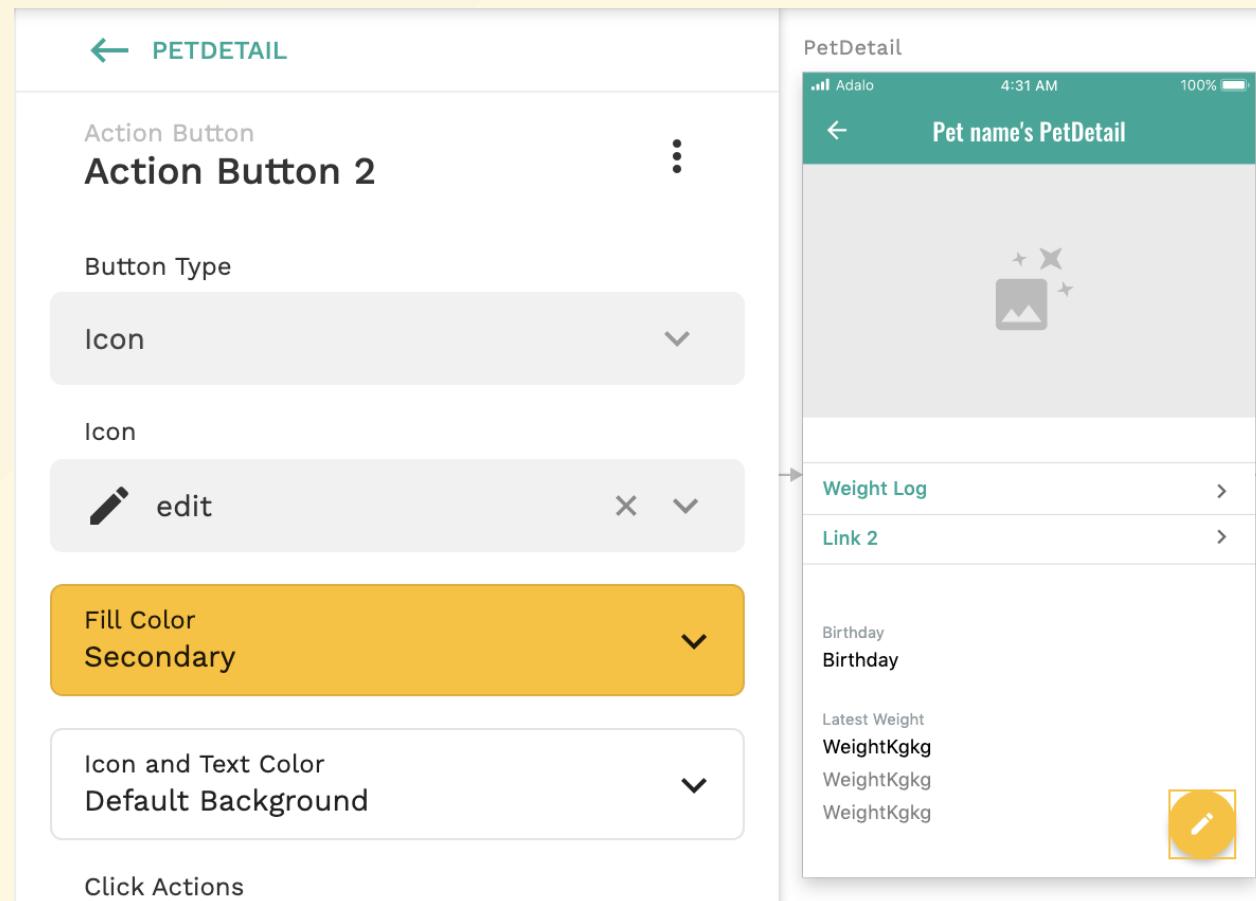
\* 体重の登録日時が省略されてしまします。同じ日に複数の体重を登録してテストをしたかったのでDate&Time型にしていますが、本来はDate型にして、1日に1回しか登録できないよう制御する方が良さそうです。



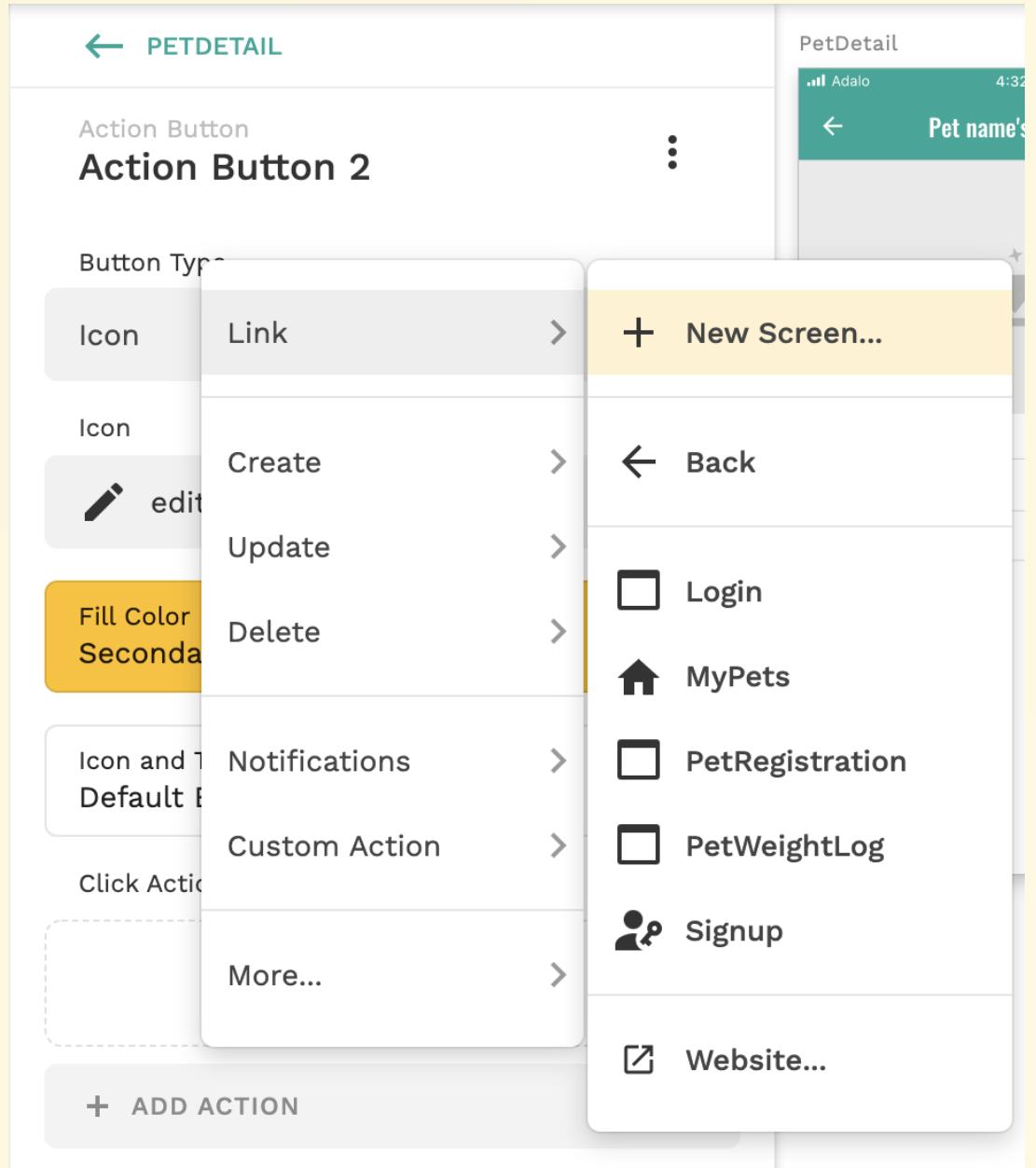
## データの更新(UPDATE)

登録したペットのデータを更新できるペット情報編集画面を新たに作ります。

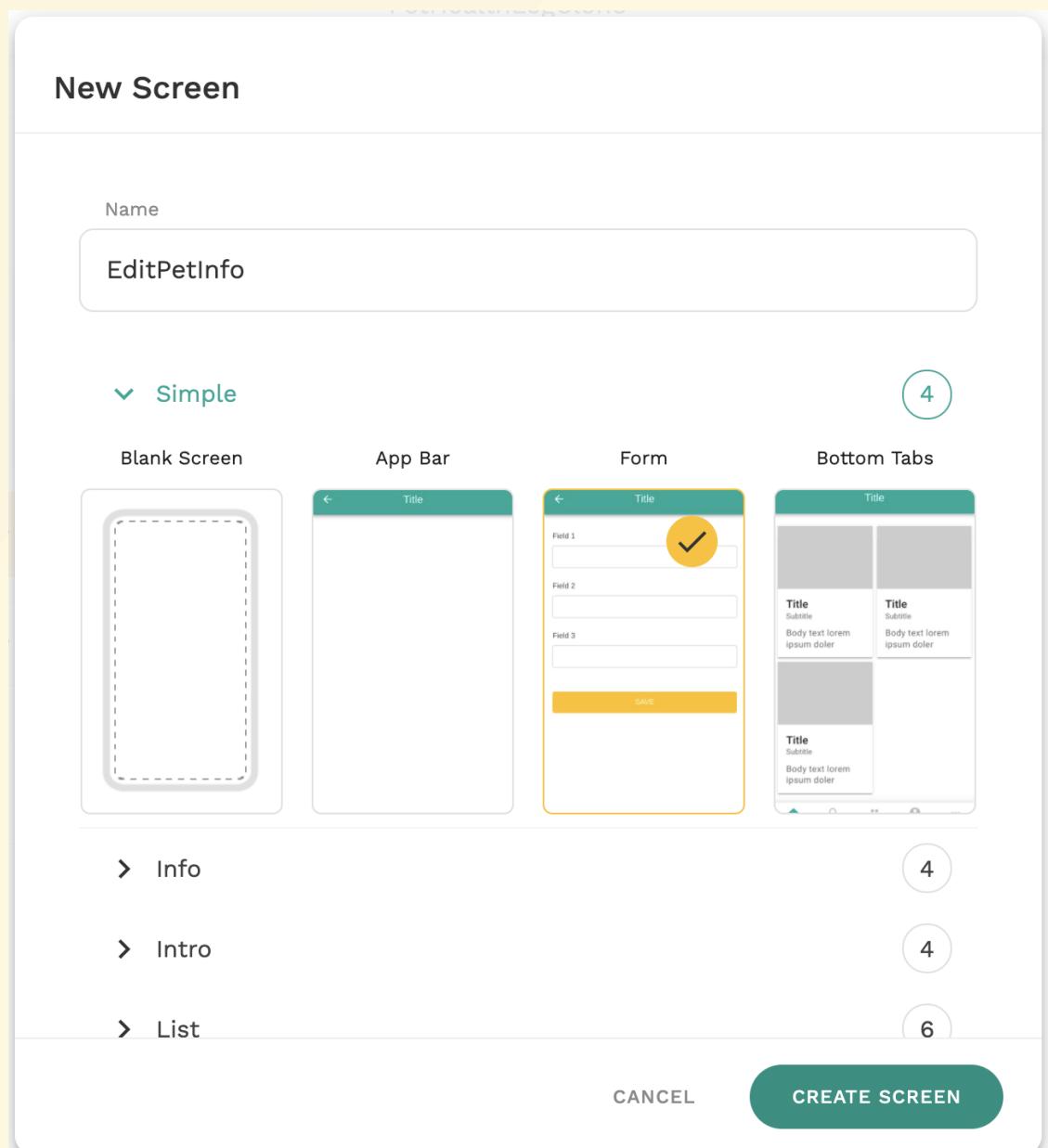
- ADD COMPONENTから Action Buttonを追加
- Iconをeditに変更
- Icon and Text Colorを Default Background(white) に変更



- ADD ACTIONからLink > New Screenを選択

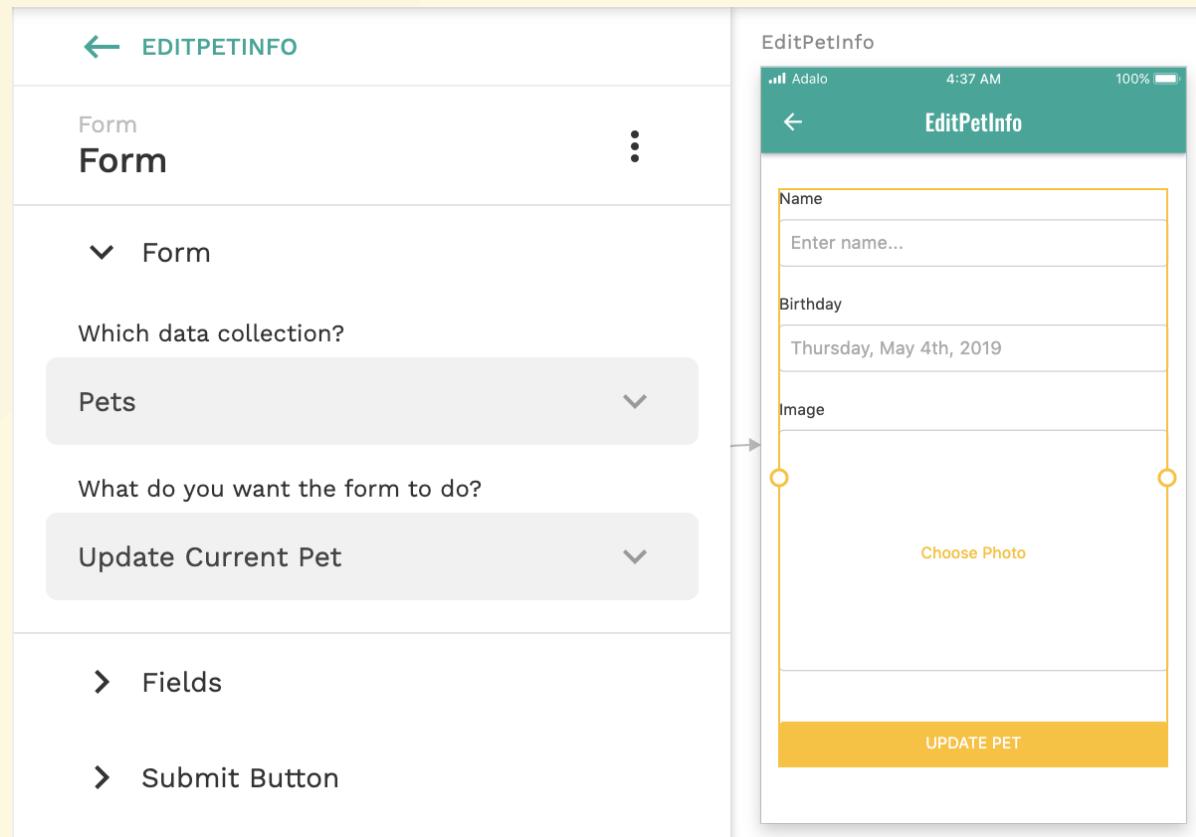


- Screen Nameを入力
- Formを選択
- CREATE SCREENをクリック

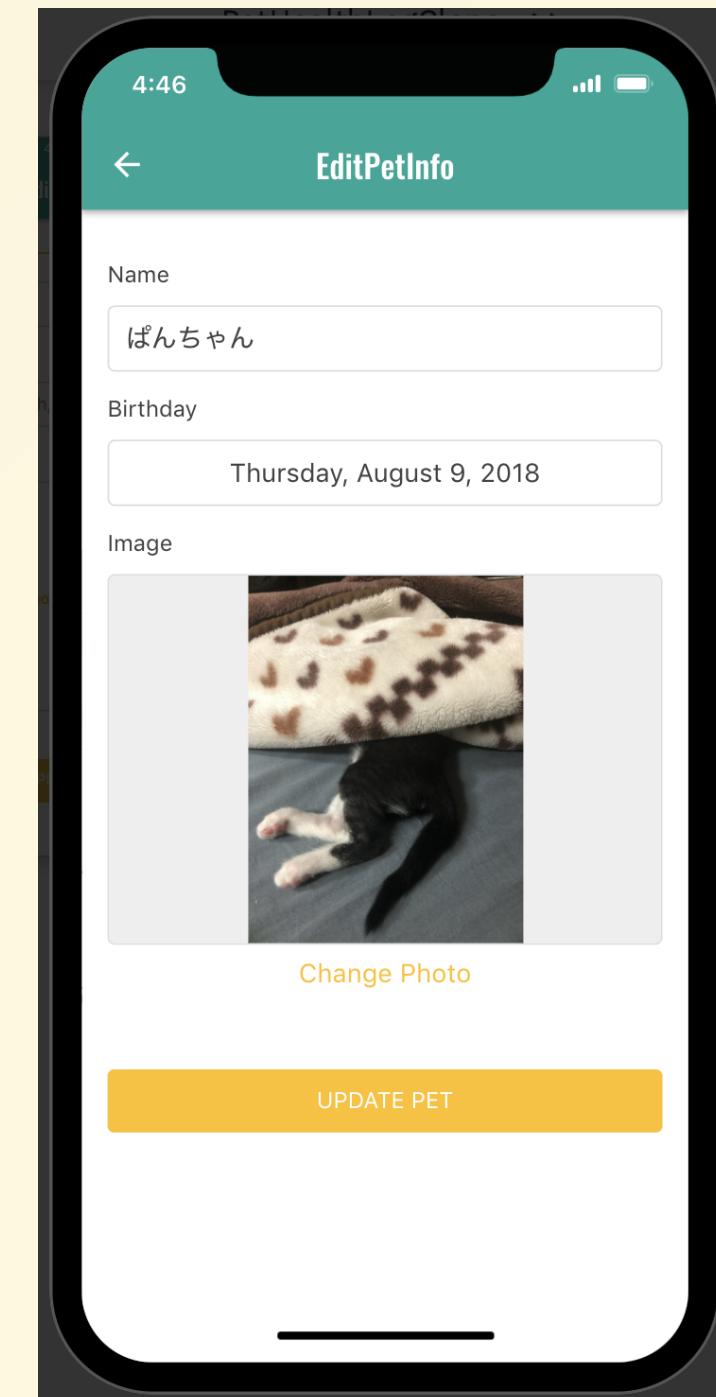


Formコンポーネントで以下の設定をするだけで、ペット情報編集画面は完成です。

- Which data collection?で Petsを選択
  - 選択したCollectionに合わせたフォームを自動生成してくれる
- What do you want the form to do?でUpdate Current Petを選択



Preview機能でペット情報編集  
画面が使えることを確認しま  
しょう。



## 補足

- 前回はペット登録画面の入力フォームをコンポーネントを一つずつ追加して作成しましたが、ペット編集画面を作ったのと同様の手順で自動生成できます。
- ペット登録画面のコンポーネントを全て削除して、入力フォームを Pet Collection から自動生成してみてください。

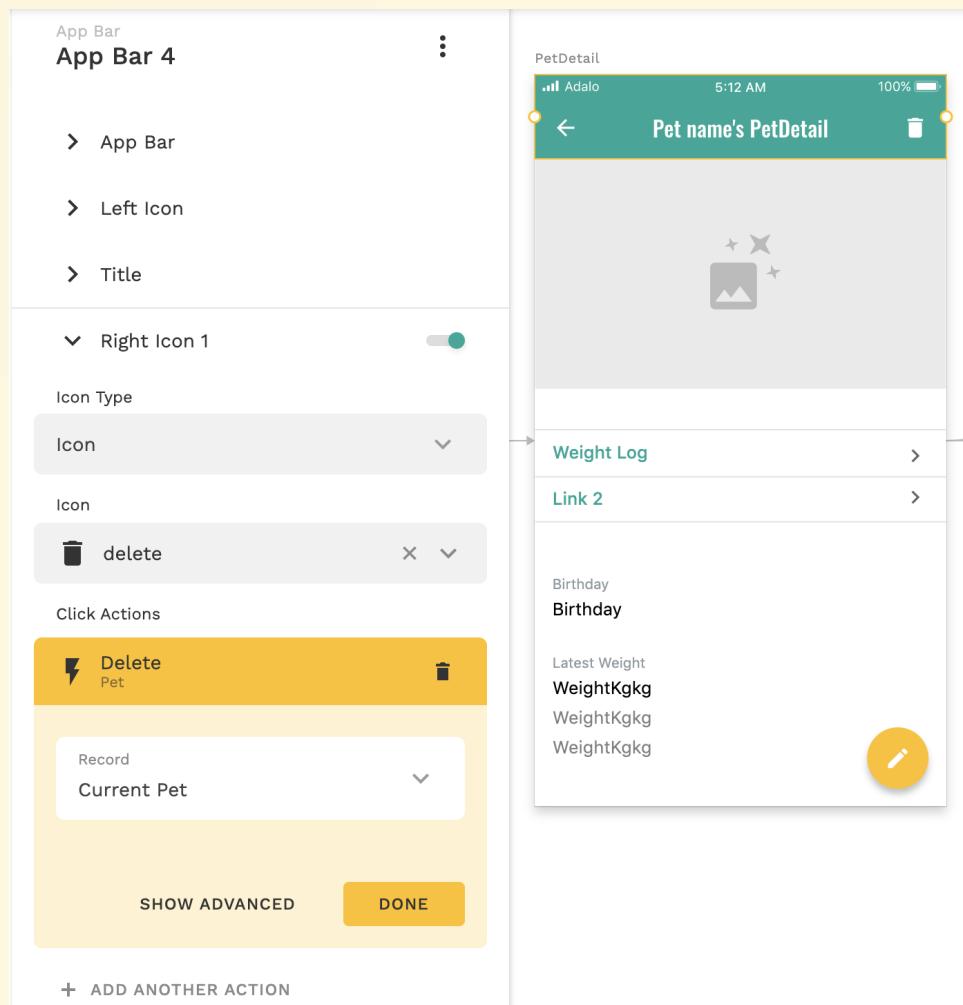
## 参考: 実際のアプリ開発の流れについて

- サンプルアプリでは、UI作成->データベース作成->それらの連携という流れで作成を進めました
  - どのような画面かがイメージできていないと、それに必要なデータが分からず、データベースの設計も難しいと考えたため、このようなレクチャーの流れにしています
- 今後Adaloを使ってアプリ開発をする際は、必要なデータが分かれているのであれば、先にデータベースの設計・構築を行うことをオススメします。それによって、UIの作成も自動生成を活用しながら簡単にを行うことができます。
  - とはいって、実際の開発は一方的な流れではなく、試行錯誤してデータベースとUIを行ったり来たりすることになります。

## データの削除(DELETE)

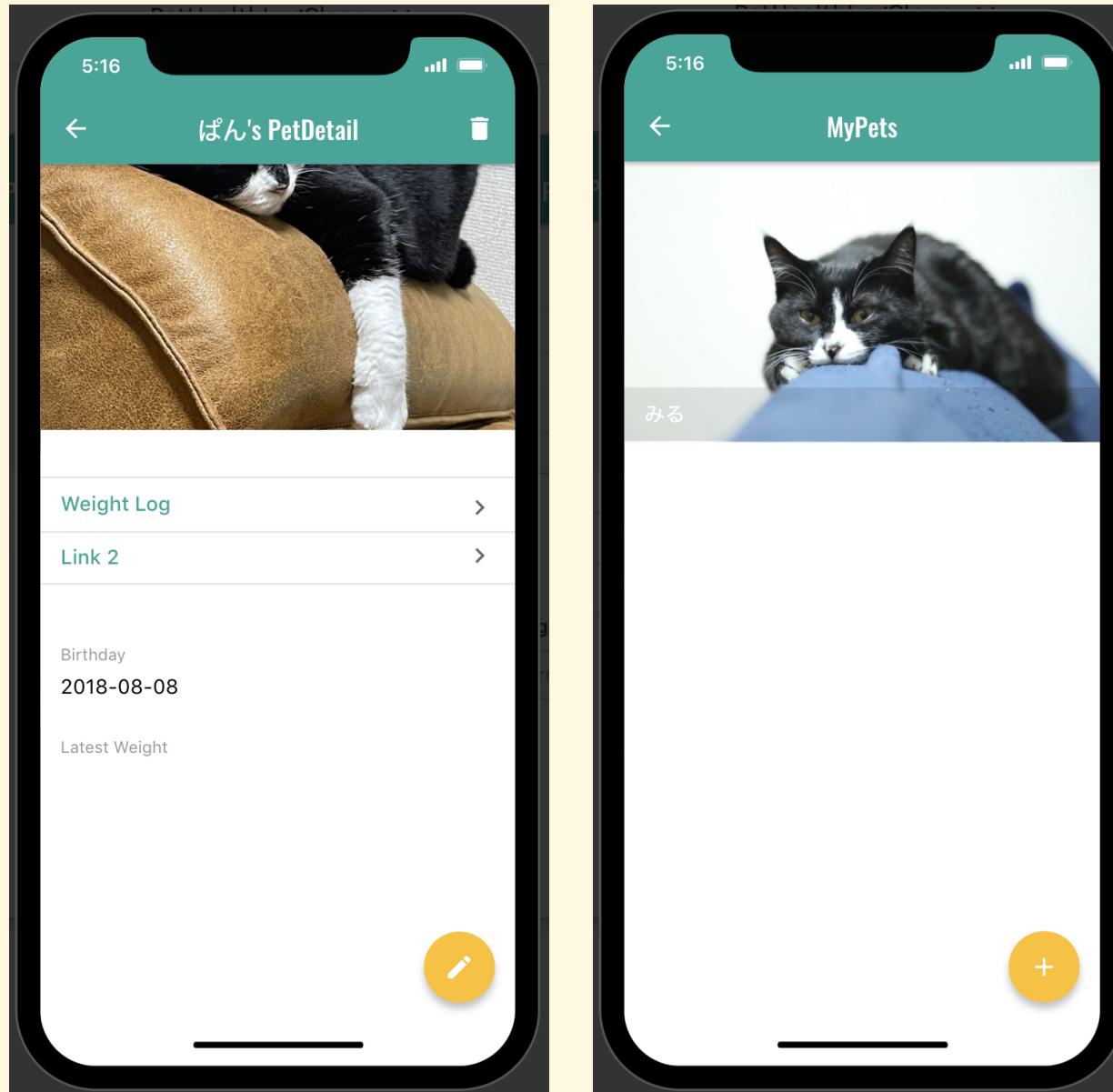
ペット詳細画面に、登録したペットを削除できるボタンを作ります。

- App Barを選択し、Right Icon 1のトグルをONにする
- Iconをdeleteに変更
- ADD ACTIONからDelete > Current Petを選択
- ADD ANOTHER ACTIONからLink > Mypetsを選択



Preview機能で削除を試してみましょう。

削除が完了してペット一覧画面に遷移すると、削除したペットは表示されません。

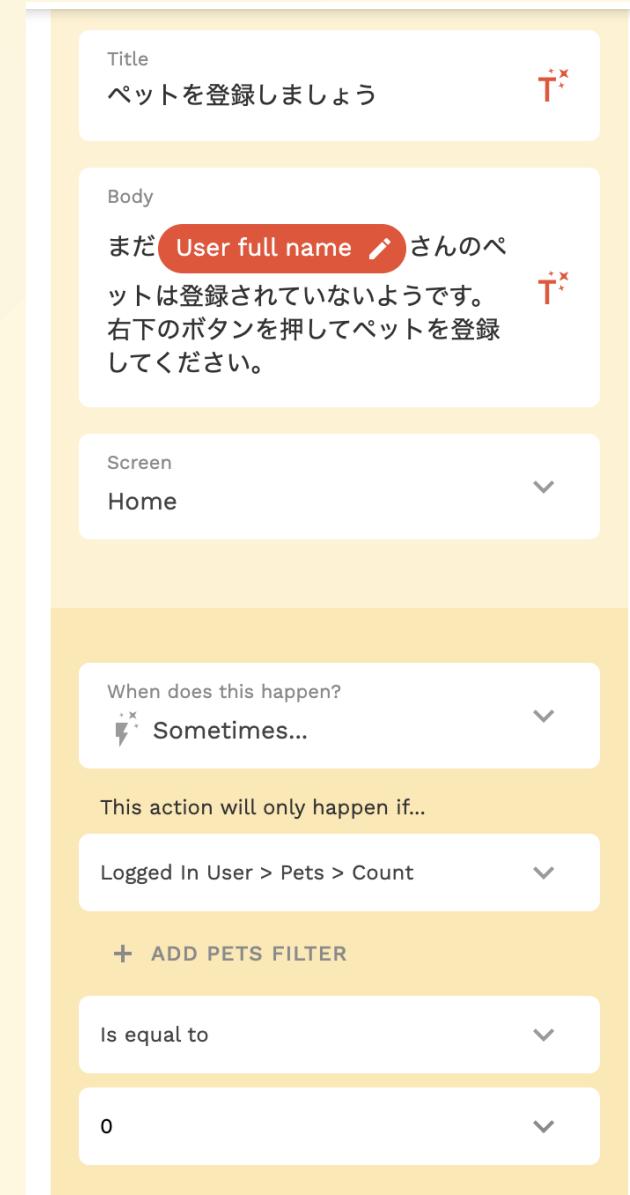


以上で、データベースに対するCRUD操作(CREATE, READ, UPDATE, DELETE)を全種類実装することができました。

# サンプルアプリを改善しよう

まだご紹介していないAdaloの機能を使いながら、サンプルアプリを改善していきます。

TODO: ペットの登録が0匹の時に登録を促すメッセージを表示させる方法の紹介



## TODO: バリデーションの紹介

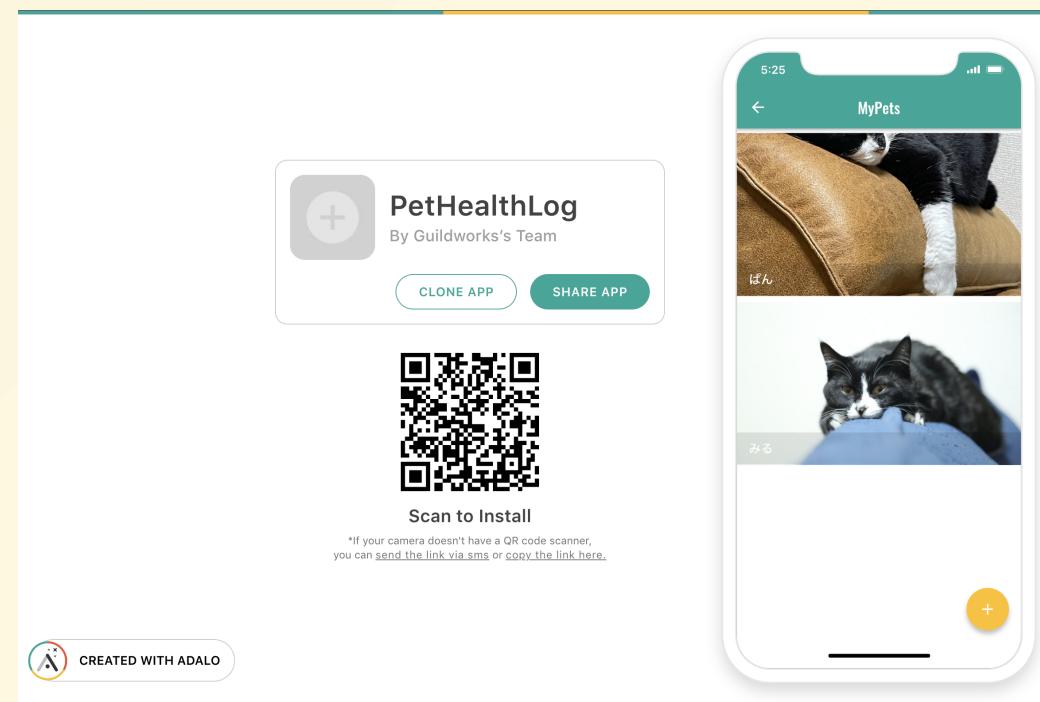
- 必須チェック
  - happen Sometimes...の使い方

TODO: その他の改善方法を追加

# クローン用URL

- 以下のURLから完成したアプリをクローンできますので、答え合わせに使ってください

<https://previewer.adalo.com/f1324ea8-ec47-4c22-a3a9-3258044eb754>



# 外部サービス連携

## TODO

- Zapier紹介
  - Slack連携?
  - Google Spreadsheet or Firestoreとの連携でレコード上限50件を回避させる?(せっかくAdaloのデータベースを学んだのに...というところはある)
  - Twitter or Instagram連携?

# 演習

Development Phaseのチームメンバーと一緒に新規のアプリを開発してください。作るアプリは自由に決めてOKです。

作るアプリが決まらないチームは、チームメンバー紹介アプリ(メンバー情報のCRUDができるもの)にしましょう。

- \* 今日のレクチャーにチームメンバーが参加していない場合は、新しくチームを組んでもOKです。
- \* 最後は全チームに発表していただきます。
- \* アプリが使えるようになったら、SlackでURLを共有して、みんなに見てもらいましょう。

## 演習についての補足

- TODO: チームメンバーのアプリ編集への招待方法を紹介
- 同時編集での注意点
  - リアルタイムで他の人の編集は反映されません。同じ画面を編集すると上書きされてしまうので、別の画面を編集しましょう。上書きが恐ければ、みんなで同じ画面を見ながら一緒に操作しましょう。

## 演習結果の発表

チームごとに演習で作ったアプリについて発表してください。

# まとめ

- TODO

**以上です！**

**お疲れさまでした！**