Advanced Mobile Lab

Developer Meetup

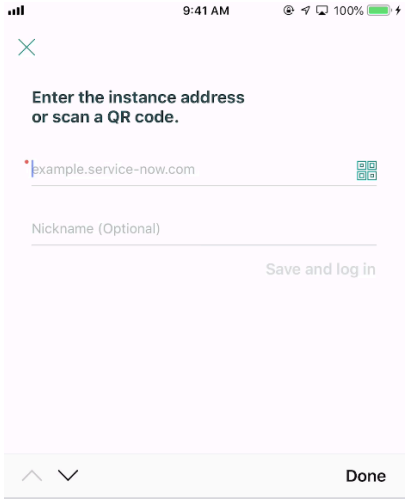
March 2019

Lab 1 – Sert up Mobile client and instance

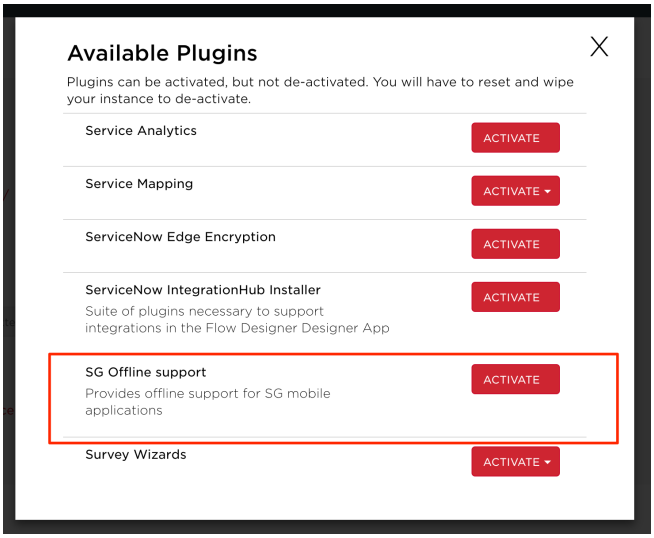
Lab Goal

In this lab, we will download Mobile client and connect it to your instance.

1. Navigate to the your Madrid based PDI and login as an Admin
2. Download the **ServiceNow Agent** app on your phone from IOS App store or Android Play store.
3. Now if you open the app, it will ask for the instance name and nickname, enter them and click **Save and login**



4. When you click Save and login it will take you to a web page for authentication, provide the instance credentials.
5. Grant account permission, if asked for.
6.
7. Activate offline plugin, Go to developer portal, **manage > instance**
8. From **Actions**, select **Activate Plugin**
9. Choose to activate **SG Offline support**

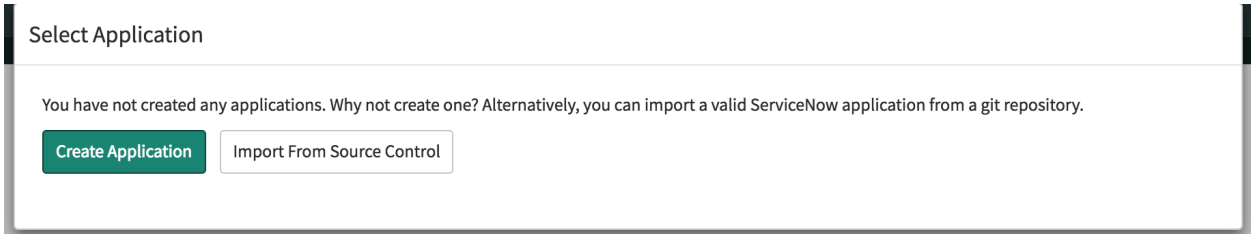


Lab 2 – Introduction to Mobile Studio and Creating a mobile application and an Applet

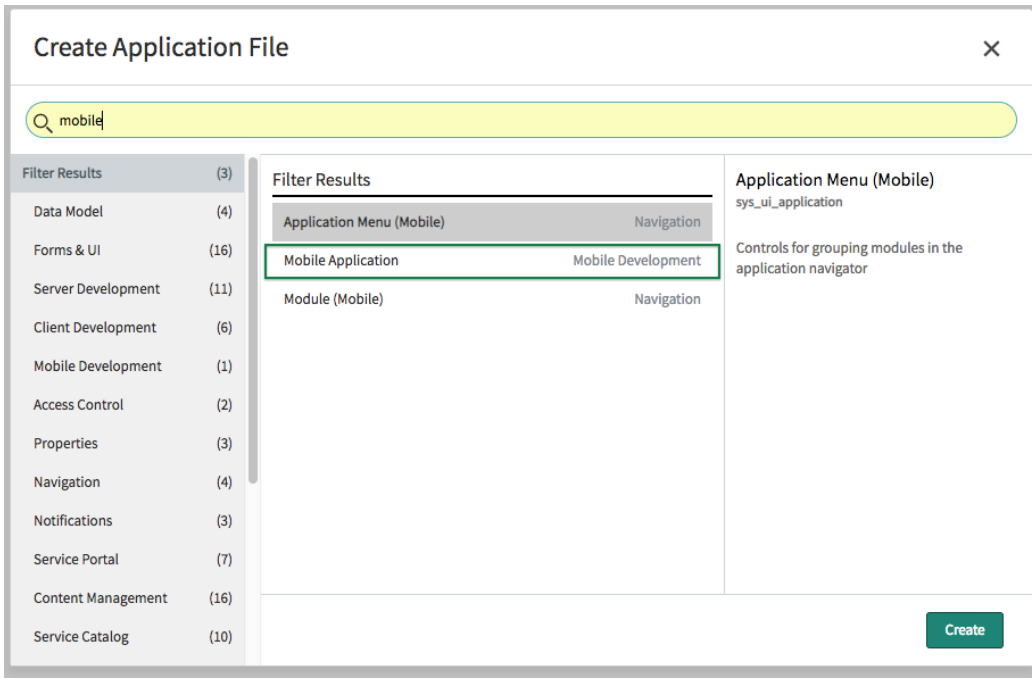
Lab Goal

In this lab, we create a new mobile application using the studio.

1. Login to your PDI (Must be Madrid release)
2. Make sure the following plugins are activated
3. Go to studio and click **on import from source control**.
4. URL: **https://github.com/arnoudkooi/sn-meetup-mobile.git** and click import. (this will create a table Claims and a few records to test)



5. After the import is done, click the **select application** and choose "**Meetup Mobile**"
6. Click on "**Create application file**".
7. Search for "**Mobile application**" and click on the Create button.

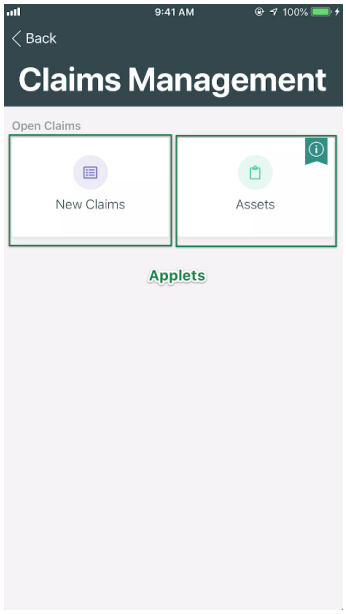


8. Let's name our mobile app "**Claims Management**".
9. Description, add a line describing the app, something like "**Application to manage insurance claims**"

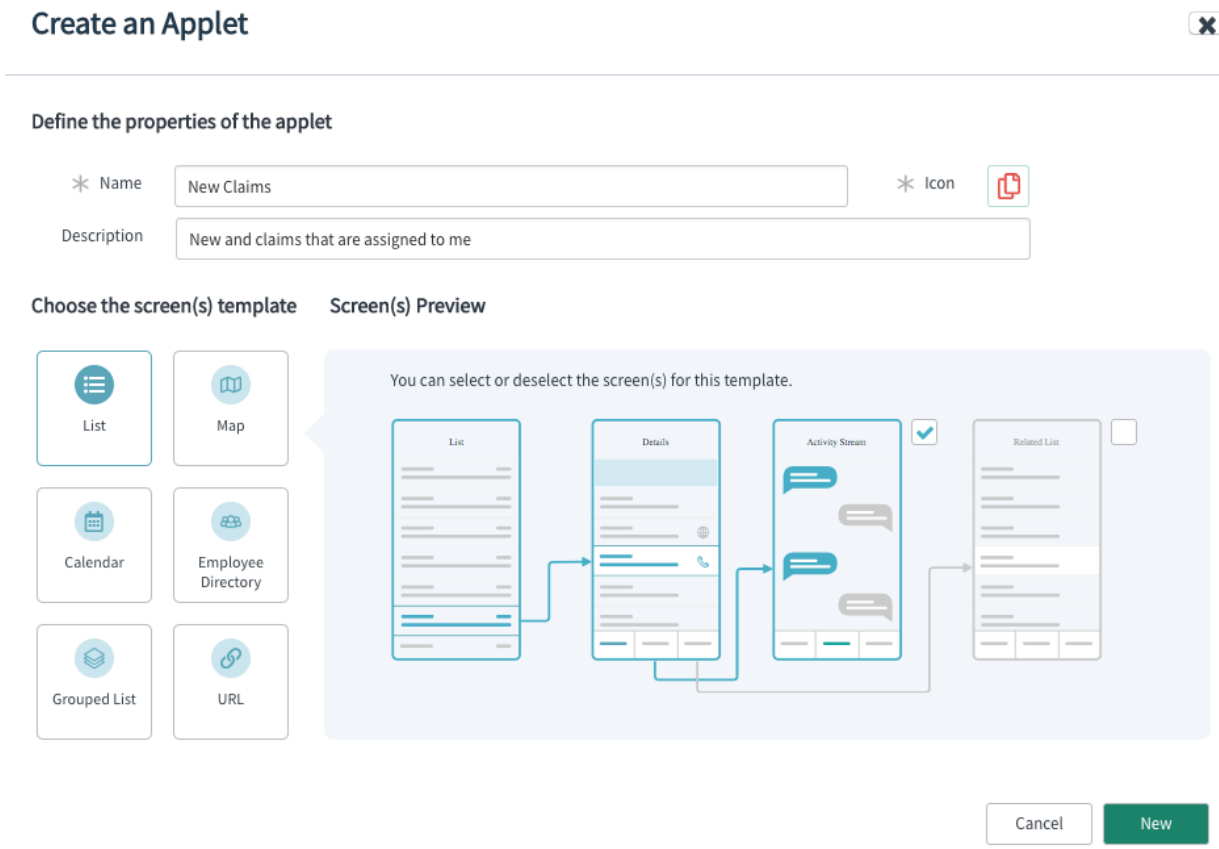
10. Choose an **icon** and **Icon color**, hit create.

11. Now that we have created our application let's start creating our **Applets.**

    Example:



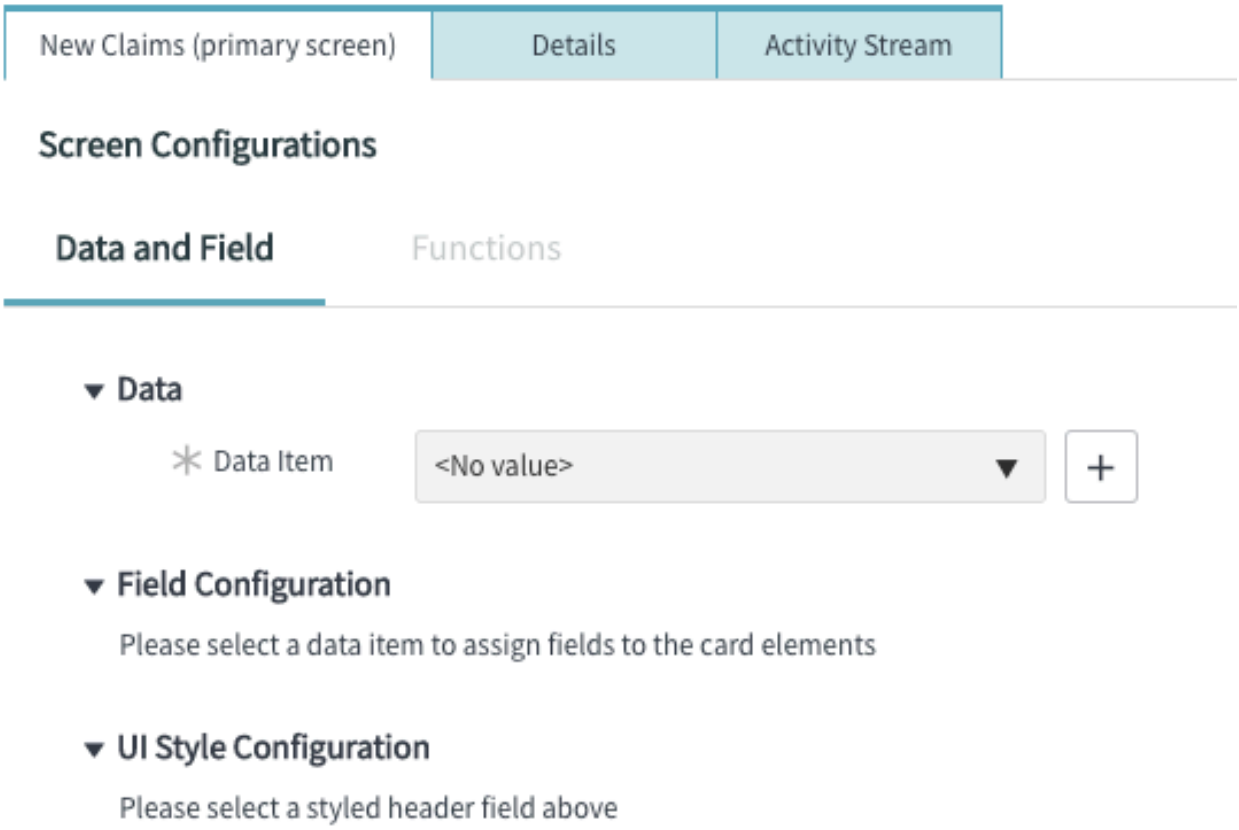
12. Click on "**Create an applet**" section
13. Our first Applet is a **List**. Let's call our applet "**New Claims**", description "**New and claims that are assigned to me**". The Description appears as ribbon on top of the applet.
14. Give it an **Icon** and **Icon color**



15. When you click on different applet Templates you can see the preview on the right side.
16. We can choose the **Activity Stream** and **Related List** to be present.
17. Since we selected the **List** type **applet**. Our initial screen is going to be a **List of Claims**
    Here is the Screen hierarchy for this applet

    List-> Detail -> Activity Stream

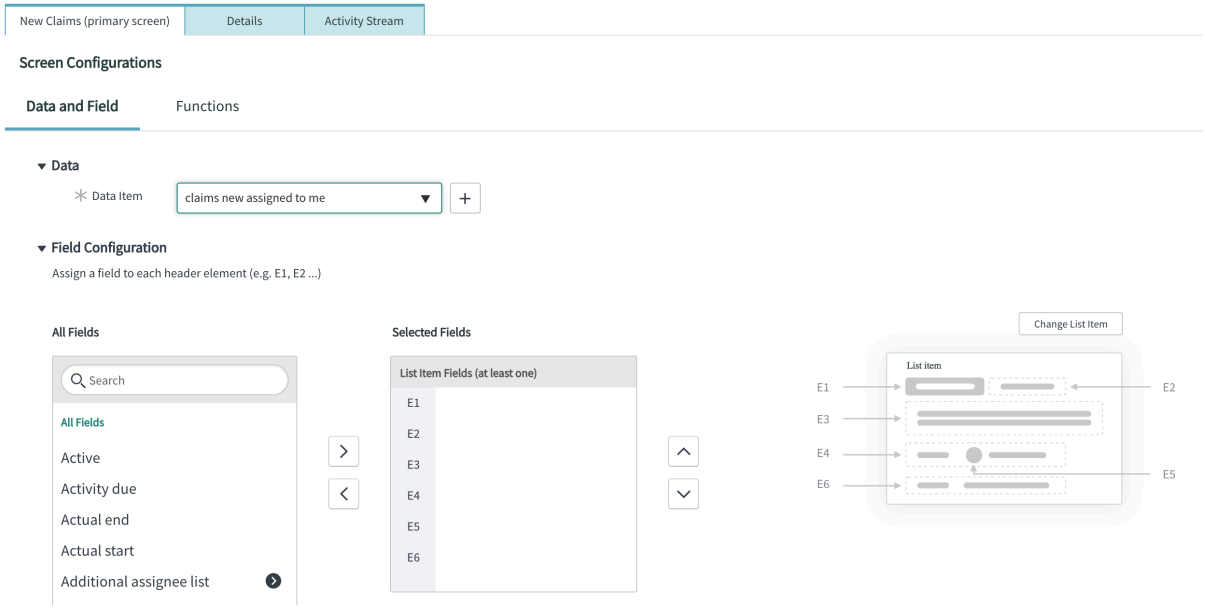
18. **Click on New.** This will take you to **Applet Configuration/Definition Screen**.
19. Here we can configure how all 3 screens (**List-> Detail -> Activity Stream**) will look and function.



20. For our List screen, we need to define where to get the Data from. That is the **Data Item** field.
21. Click on the **+** plus icon to create a new Data Item.
22. Let's call our data item "**claims new assigned to me**" and in our app, for table choose **Claims** table.
23. Give it some description (Optional)
24. Configure the Query Condition to display all **Claims** that are in **State -> New OR Assigned to is (dynamic) me.** Save the data Item.

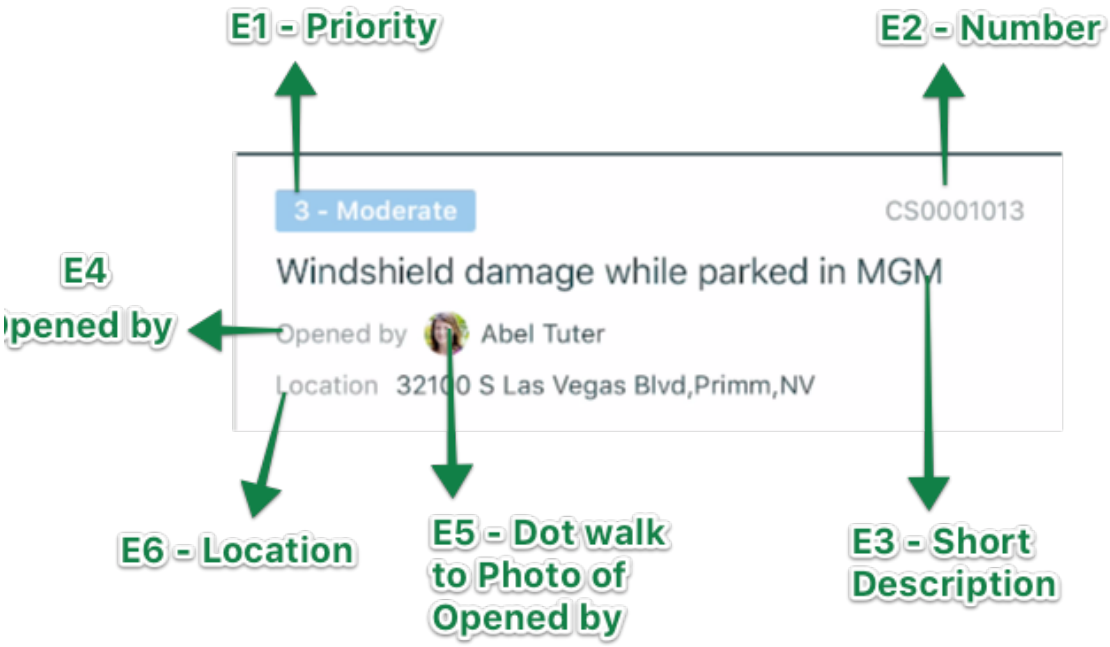
TIP: Give items in the backend lowercase and descriptive names. For data item with tablename with query condition description. This keeps it easier to manage as your app grows.

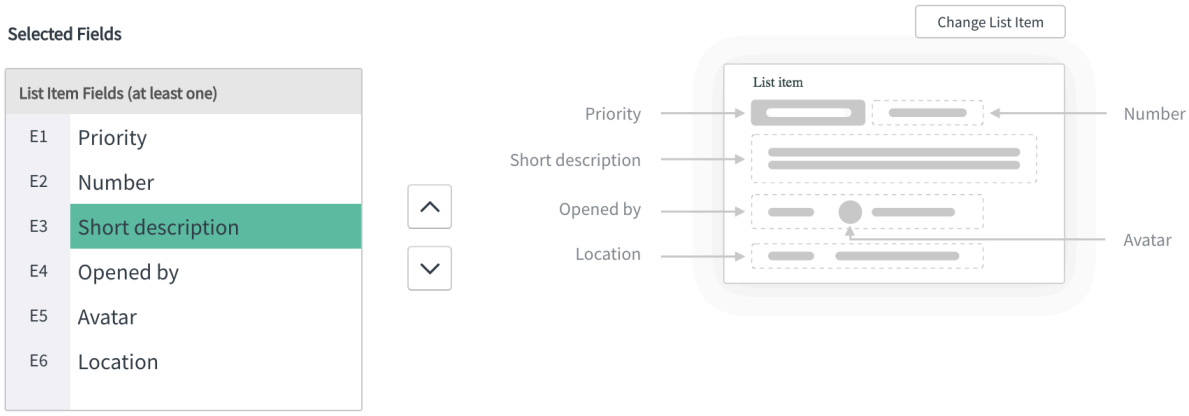
25. Assign this **Data Item** to the Applet. As soon as you assign the data item, you should see the Fields from Claims table appear to configure the List View.



TIP: You can change the List view by click on the "Change list item" button. We will use the default one.

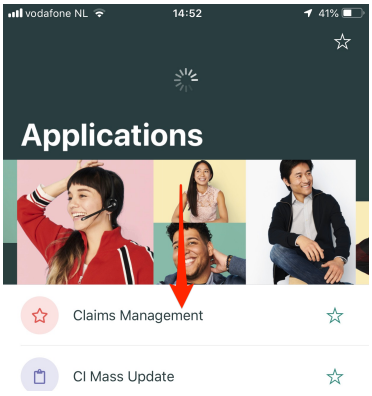
26. Start Choosing the fields for E1, E2, E3 etc. In our Case here is the field mapping

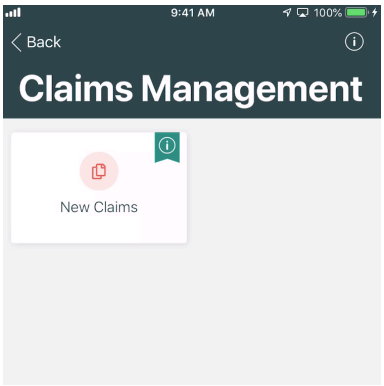


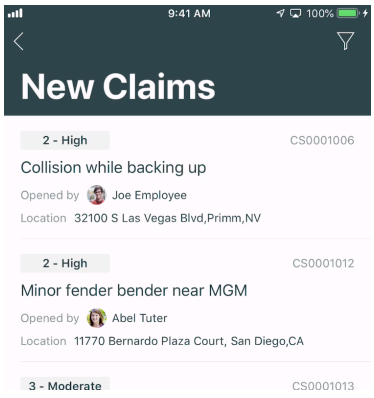


27. Save the screen.
28. Time to test out what we have so far.
29. Go to the start page in the app, swipe down to refresh

Verify: You should have New Claims applet and List of Claims







TIP: Always swipe down from the start page to refresh the app.

You can change the applet properties like Name, Description, offline, Applet Template type, enabling or disabling the Activity Stream and Related List pages by clicking on the Properties button.

Lab 3 – UI Styles, Detail and Activity Stream

Lab Goal

In this lab, we will configure the Detail screen and add UI Styles.

Our list looks a little dull without any colors, let's add some colors to **Priority Field using UI Styles.**

1. In Studio left-hand nav launch the UI Styles by clicking pop-out icon.

2. Click on the new button to create UI Style. We will create 4 new UI styles for our applet.

1st UI Style for Priority critical

Name	claim priority critical
Table	Claim
Condition	Priority is Critical
Field Name	Priority
background_color	#C83C36
font_color	#FFFFFF

2nd UI Style for priority High

Name	claim priority high
Table	Claim
Condition	Priority is High
Field Name	Priority
background_color	#FE9741
font_color	#FFFFFF

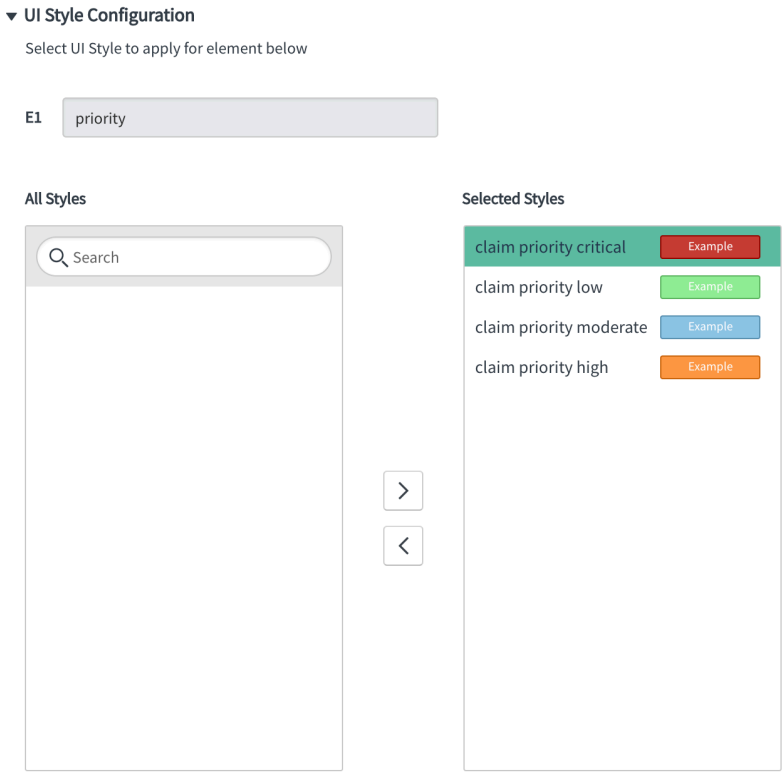
3rd UI Style for priority Moderate

Name	claim priority moderate
Table	Claim
Condition	Priority is Moderate
Field Name	Priority
#8dc3e7	#8dc3e7
font_color	#FFFFFF

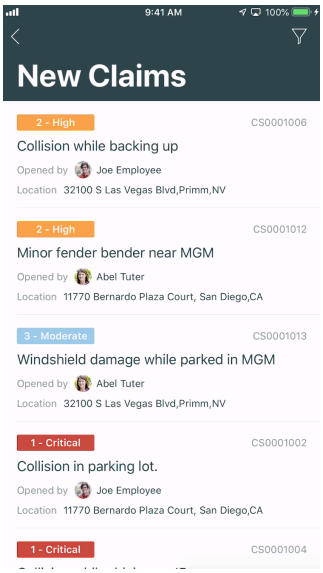
4th UI Style for priority Low

Name	claim priority low
Table	Claim
Condition	Priority is Low
Field Name	Priority
background_color	#92ED96
font_color	#FFFFFF

3. Navigate back to New Claims applet screen. If you scroll down, you should see our UI styles.
4. Move them to Selected bucket and **save**.

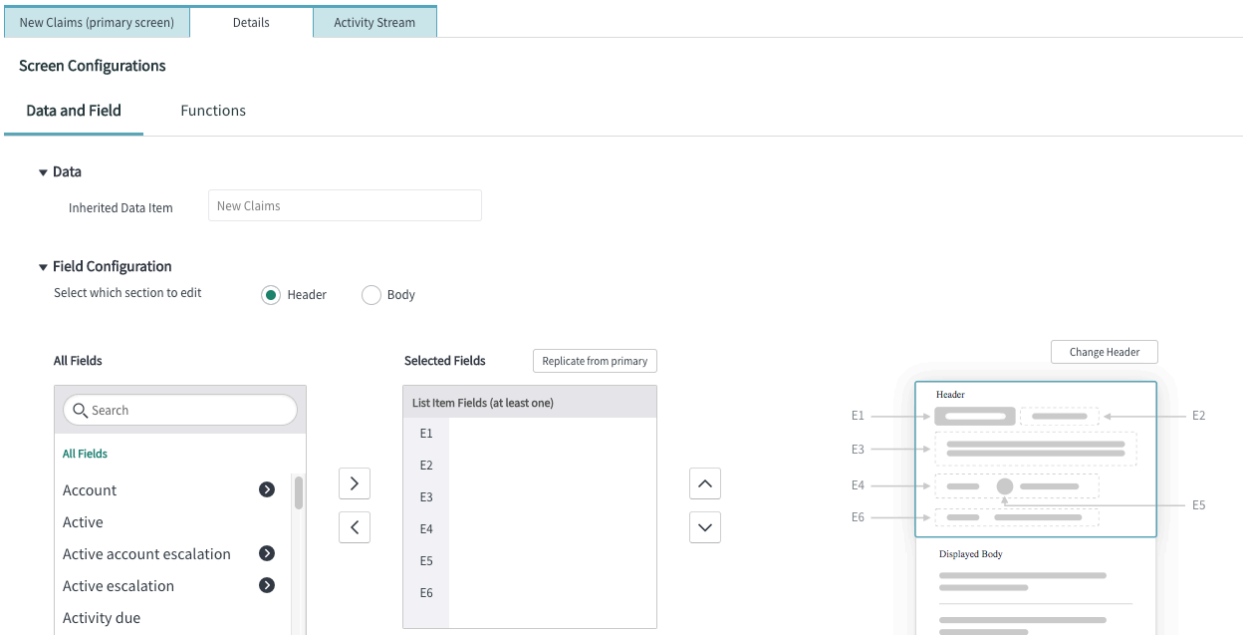


5. Verify the app to see if the UI styles were applied correctly.



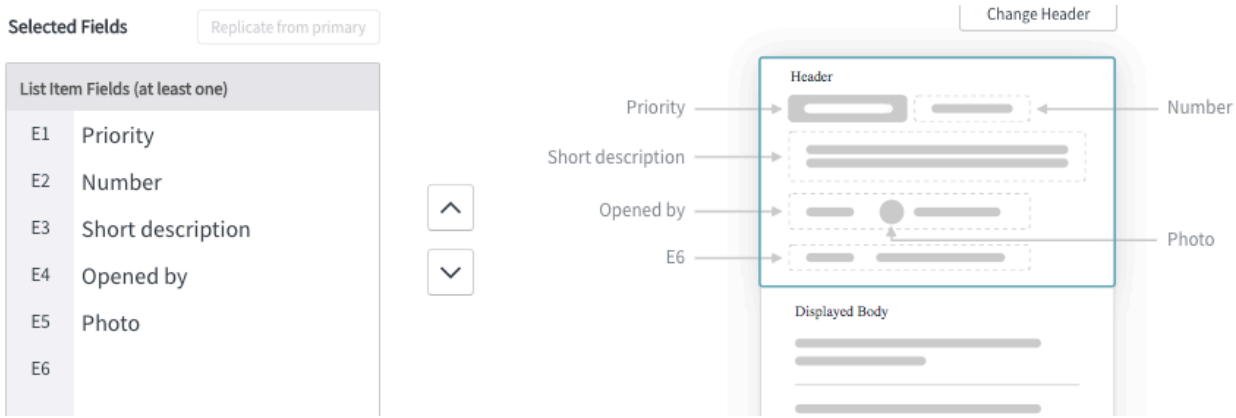
Configure Detail Screen

1. Navigate to **Details** Tab of New Claims Applet.
2. We configure the Header portion of the Details screen now.

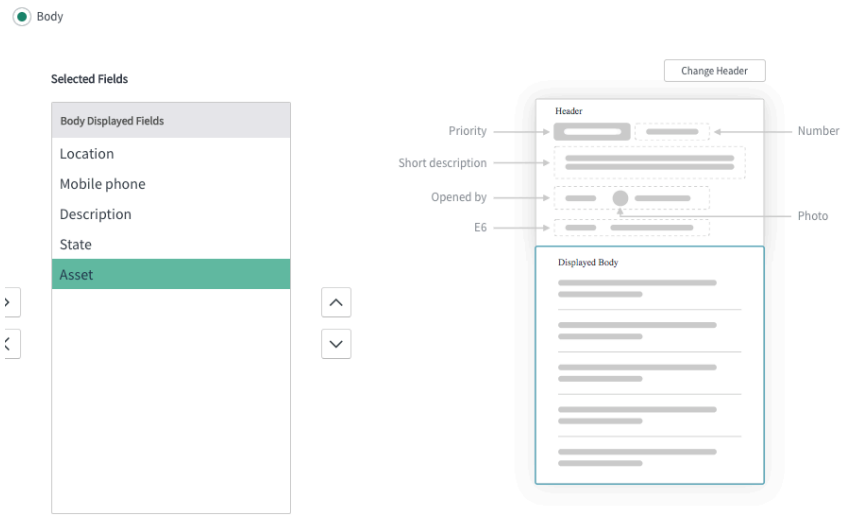


TIP: If you want the same header fields as previous primary screen click on Replicate from primary button. We can also go ahead add all the fields one by one for the header.

3. Click on "**Replicate from primary**". We will **remove the Location Field** from the header for now**.**

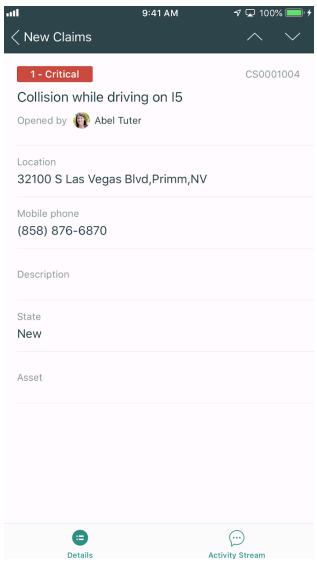


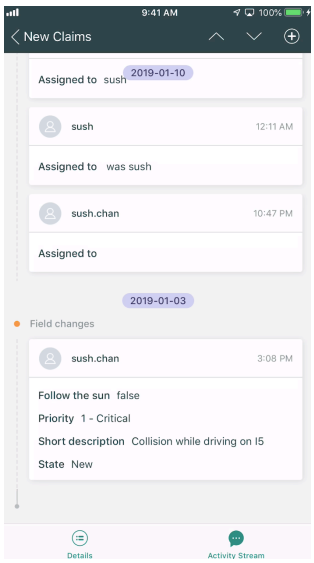
4. Scroll down to add the UI styles to Selected Bucket and **save**.
5. Now that we have configured Header, toggle the radio button to configure "**Body**"
6. Add these fields to body **Location, Mobile Phone (Opened by), Description, State, Asset** and **Save**.



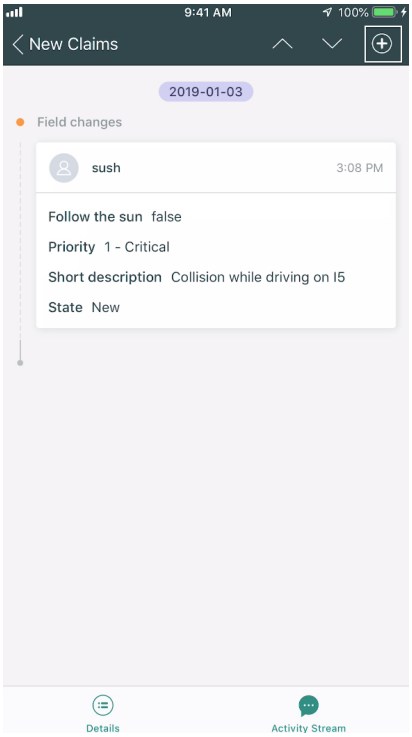
Click on **Activity Stream tab** to view the options available on Activity stream screen.

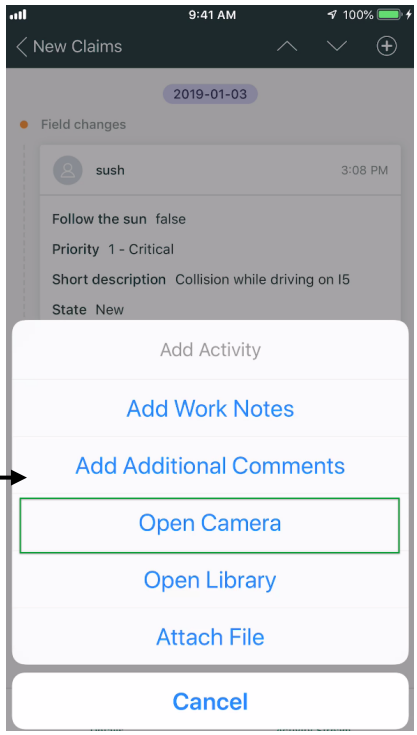
Verify the Changes: Good looking details screen and let's take some photos and upload it to the claim.

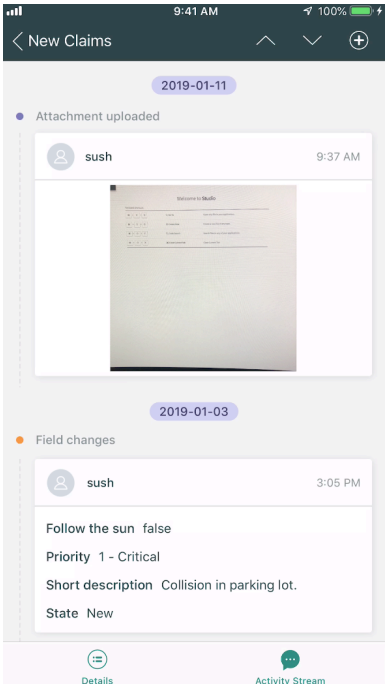




On the activity stream, click on + icon on the top right corner. You will get a list of options, click on Open camera and take a photo. It will be added as an attachment to claim record and will appear on the activity stream







Lab 4 – Smart Functions

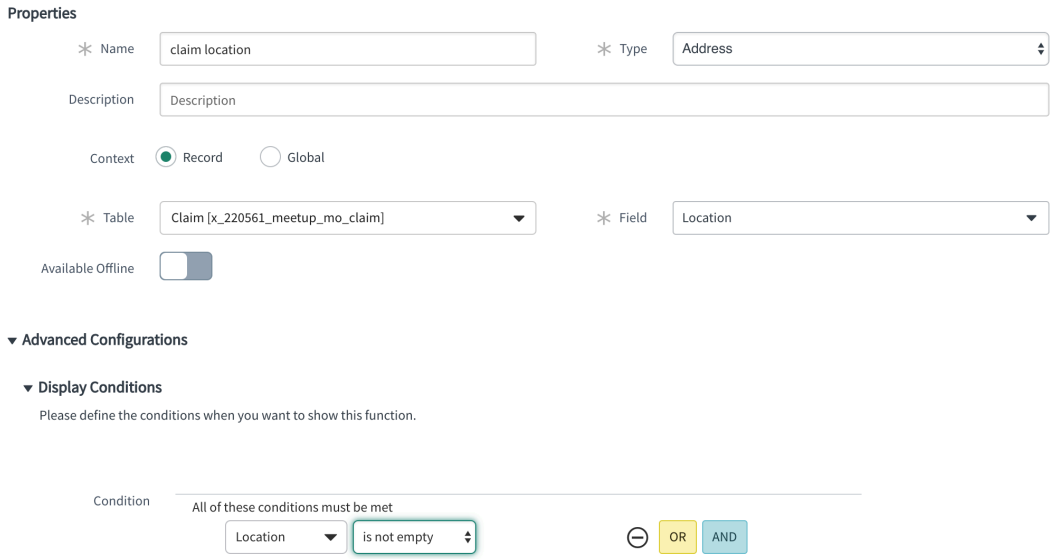
Creating a new Smart Button called Maps.

1. We can launch Navigation apps like Google maps, Waze, etc. using Smart Buttons
2. Launch Smart Buttons by clicking on pop-out icon.

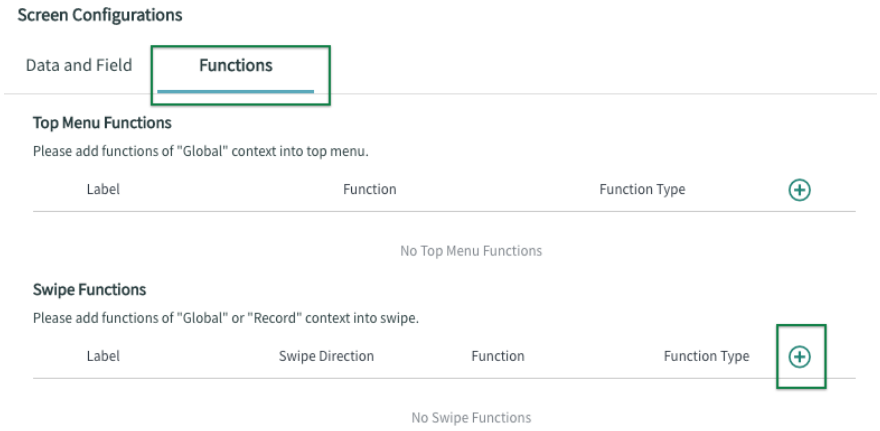
3. Click on the New button

Name	claim location
Type	**Address**
Context	**Record**
Table	**Claim**
Field	**Location**
Available offline	**False** (doesn't make sense to launch maps when we don't have an internet connection)
Condition	**Location is not empty**

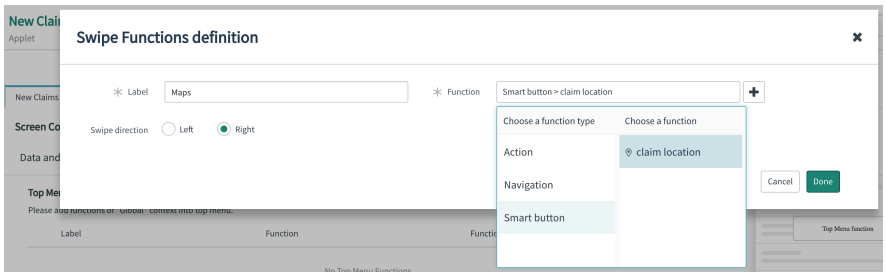
4. **Save** the record.



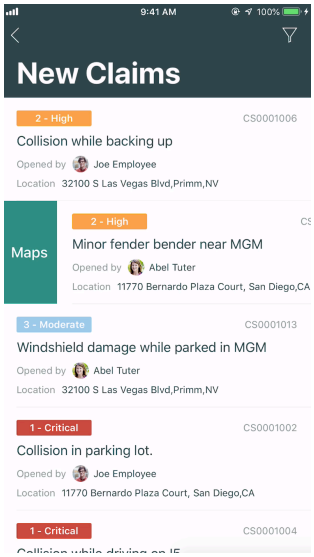
5. On **New Claims** applet screen and click on Functions tab.
6. Add a new swipe function by clicking on + plus icon

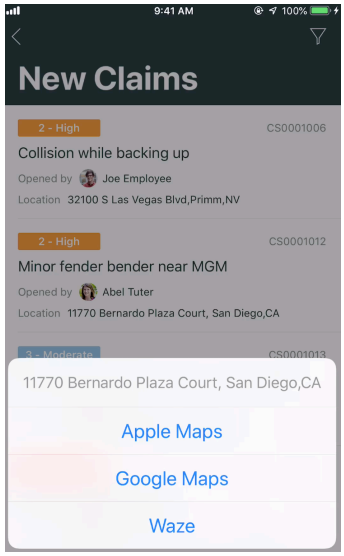


7. Create a new Swipe function record like below. Choose the newly created Smart Button as function. You can choose any swipe direction, I have the swipe direction as right. **Save the applet**



8. **Verify**: Check if swipe action works.





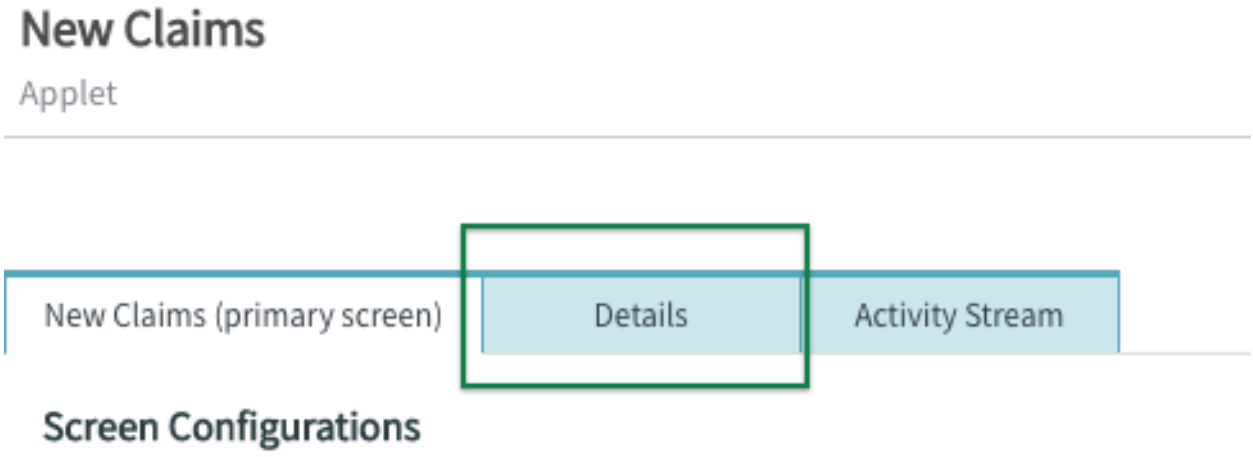
Creating a new Smart Button, add it as Field function to phone.

1. Launch Smart Buttons by clicking on pop-out icon.
2. 

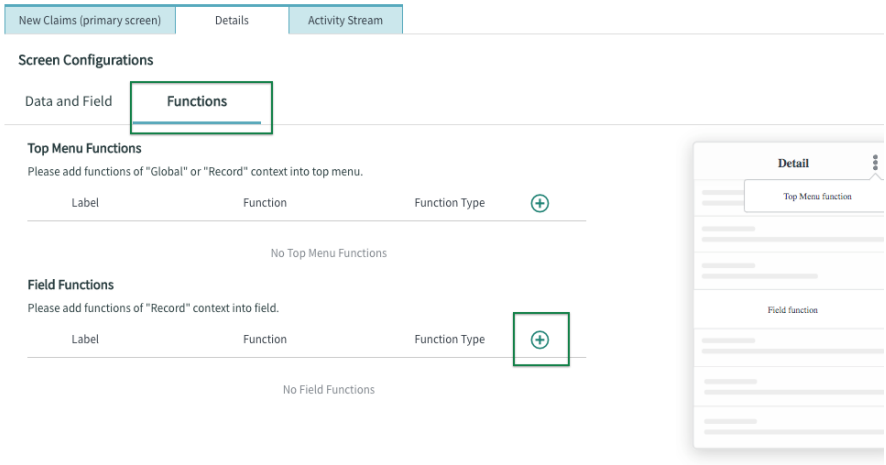
3. Create new

Name	claim - call requestor
Type	**Phone**
Context	**Record**
Table	**Claim**
Field	**Opened by -> Mobile Phone**
SMS (optional, we can add a default text for SMS)	**Hi, this is Joe from xyz insurance, I am on my way to verify your claim. Thanks**
Available offline	**True**
Condition	**Opened by is not empty**

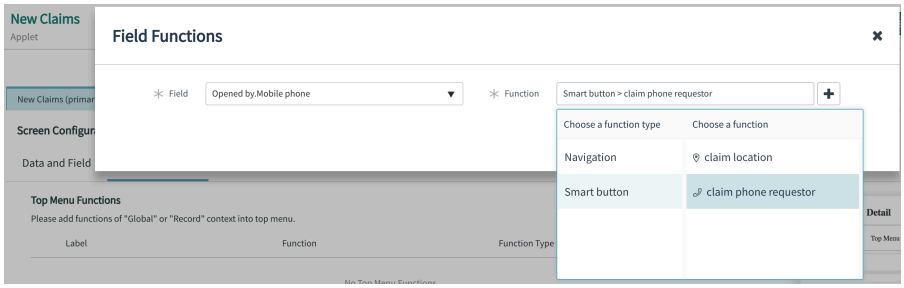
4. Save the record
5. Open **New Claims** applet screen and click on the Functions tab.



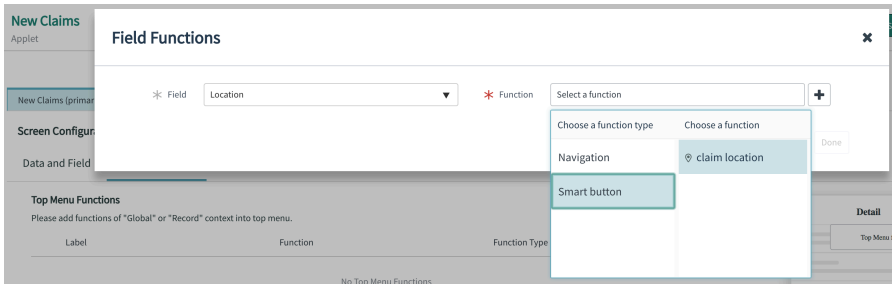
6. Click on function tab and add a new Field function



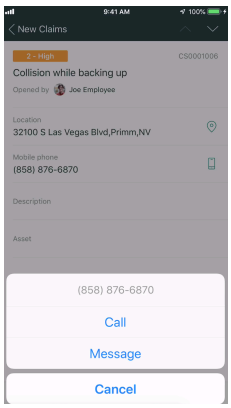
7. Add a field function field for **Opened By. Mobile Phone field**

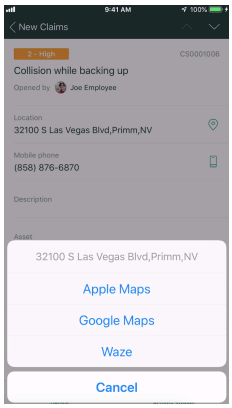


8. While we are here, let's add another field button for Location field.
9. Click on + icon and add the field function



10. **Save** the applet
11. Verify: If the two Smart buttons are working on details screen.





Lab 5 – Actions

Creating a new Action called Assign to me.

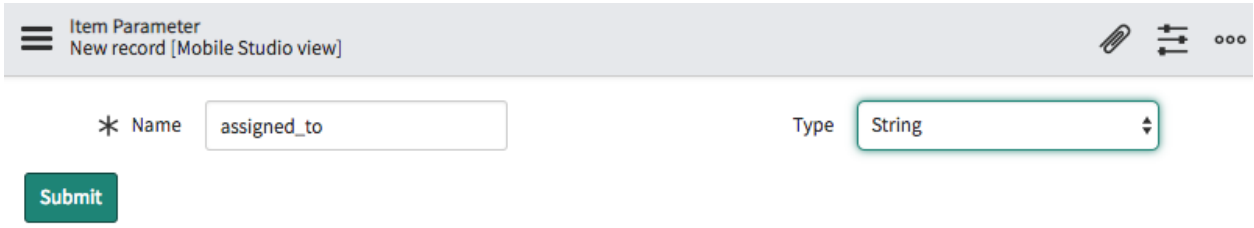
1. For an action function to work, you need to create an **action item** to associate with the action function. Action items define what the action function is and how it works.
2. Let us create an **action Item** first. Launch the Action item by clicking on the pop-out icon.

3. Create a new Action item.

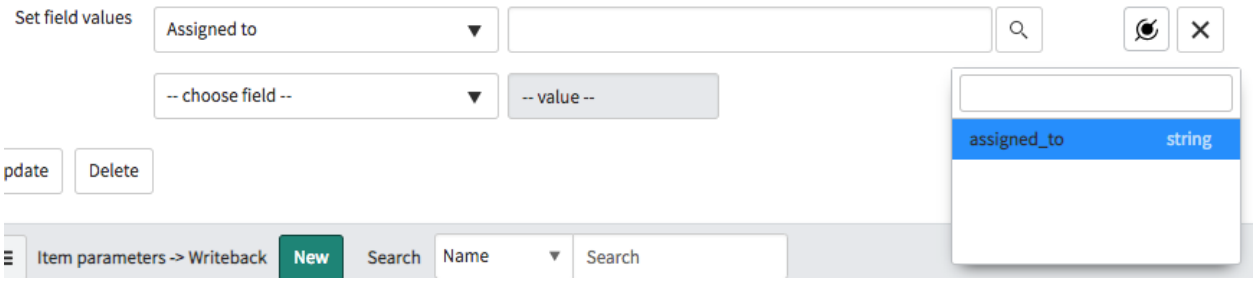
Name	claims assign to me
Type	**Update**
Table	**Claim**
Use current record as condition	**True**

4. **Save the Action Item.**

5. Now we need to get **Current User**, so we can set Assigned To field value above.
6. Let's Create a new **item parameter** variable so that we can pass the Current User to this variable in later steps.
7. **Parameters are a way of creating a variable or a placeholder that is waiting for input from either the user or the database.**
8. Create a new Item Parameter by clicking on the new button in the related list.
9. The Call is **assigned_to** and type be **String and save it.**



10. Go back to our Action Item "**Claim assign to me**", **reload the related list** to make sure the Item Parameter we added above shows up in related list.
11. **Set field values** Choose **Assigned to** click on this icon  and choose **assigned_to and save.**



12. Now that we have **action item** ready, let's create the **Action**
13. Launch Action by clicking on the pop-out icon

Action details

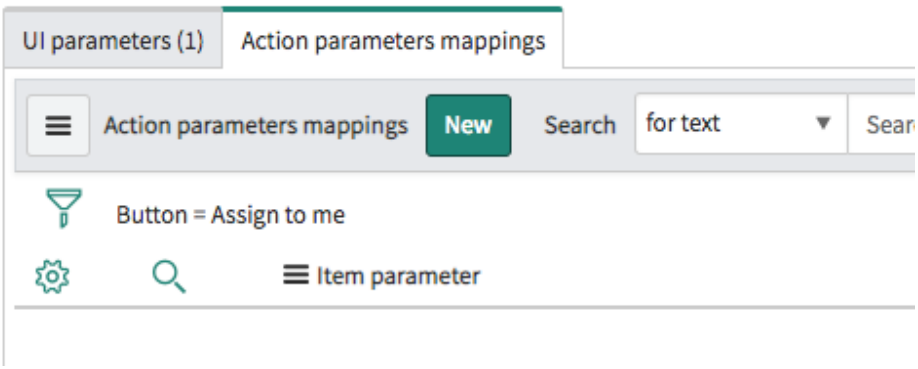
Name	Assign to me
Context	**Record**
Condition	**Assigned to is Empty**
Table	**Claim**
Action item	**claims assign to me**
Offline	**True**

14. Save the Record.
15. Two related lists called **UI Parameters** and **Action parameters mapping** appears.
16. We create a new **UI Parameter to get the current user.** Will use this to pass to action item parameter we created above in step 13.

17. Click on new to create **UI Parameter**

Name	Assign to me
Parameter type	**Button**
Input source	**Autofill**
Input type	**User**

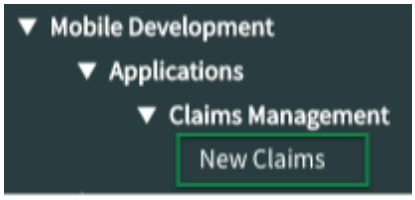
18. **Save the record**
19. Go back to "**Assign to me**" Action, by clicking on it in Studio left nav.
20. Click on **Action parameters mapping** related List.



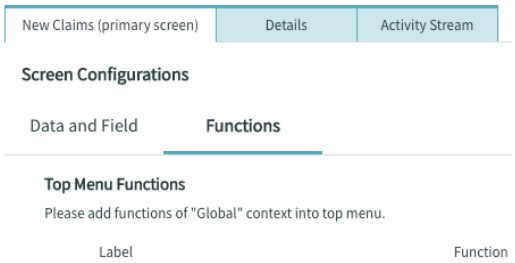
Create new

Item parameter	**assigned_to (Created on step 13)**
UI Parameter	**Assign to me (Created on step 21)**

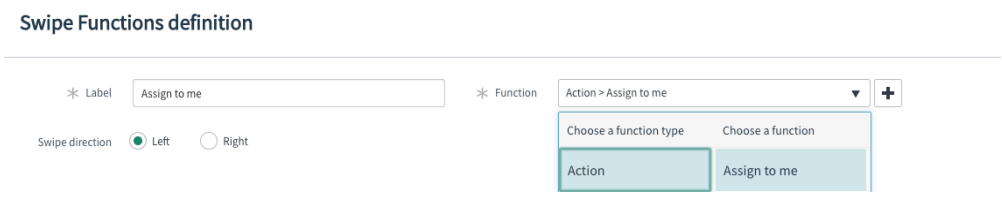
21. **Save the record**
22. Now we need to assign this button to UI
23. Launch our applet "**New Claims**"



24. Click on functions tab

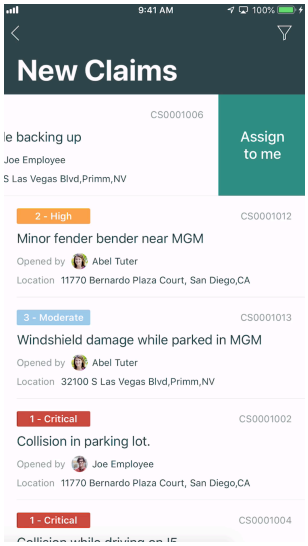


25. Add a new swipe function by clicking on + Plus icon and add the definition like below and **Save**



Save the applet too

26. **Verify: Check your app to see our Assign to me action is working correctly.**



Creating a new Action called Edit to edit the Claim.

1. Like any other action let's start by creating an Action Item called Edit
2. Let us create an **action Item** first. Launch the Action item by clicking on pop-out icon.

3. Create a new Action item.

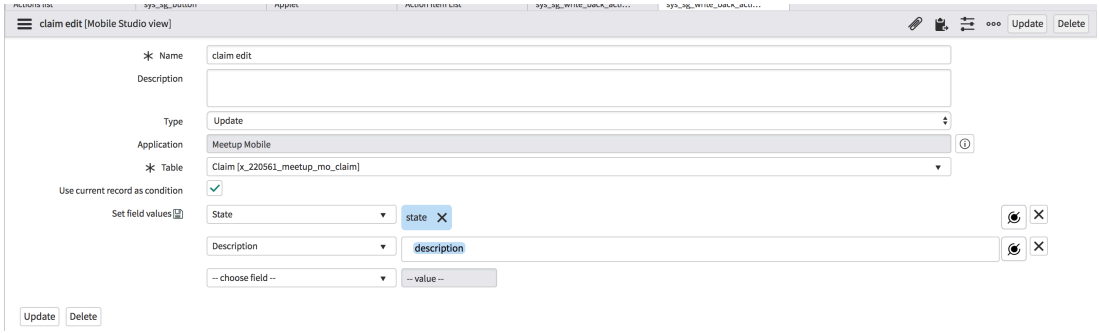
Name	Edit Claim
Type	**Update**
Table	**Claim**
Use current record as condition	**True**

4. **Save the Action Item.**
5. Now we need to get new **State and Description** fields from the user, so we can set them for the claim record.
6. Let's Create a new **item parameter** variable, so that we can pass the **State and Description** from UI to these variables in later steps.
7. **Parameters are a way of creating a variable or a placeholder that is waiting for input from either the user or the database.**
8. Create a new Item Parameter by clicking on the new button in the related list.
9. One for **description**

10. One for **State**

11. Let's assign these to **Action Item** using this icon

12.



13. Now that we have **action item** ready, let's create the **Action**
14. Launch Action by clicking on the pop-out icon

Action details

Name	Edit Claim
Context	**Record**
Table	**Claim**
Action item	**claim edit** (we created this on step 15 above)
Offline	**True**

15. Save the Record.
16. Two related lists called **UI Parameters** and **Action parameters mapping** appears.
17. Click on new to create **UI Parameter (we need to create 2 UI Parameters, one for every Item parameter we created in steps 13, 14)**

For Description

Name	Description
Parameter type	**Button**
Input source	**User Input**
Input type	**Text**
Default value type	**Source field** (if there is any value currently, show that)
Button parent table	**Claim**
Field	**Description**

For State

Name	State
Parameter type	**Button**
Input source	**User Input**
Input type	**List**
Table name	**Claim**
Field name	**State**

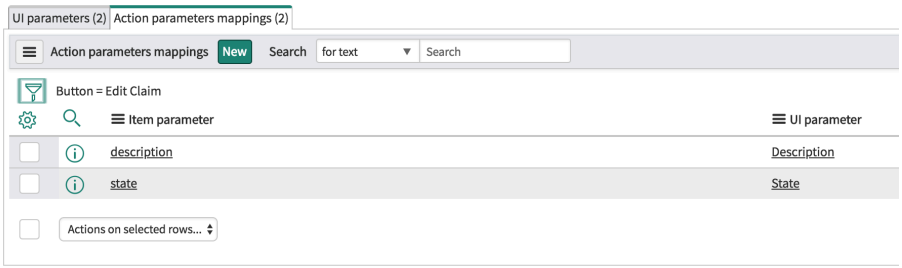
Default value type	**Source field** (if there is any value currently, show that)
Button parent table	**Claim**
Source field	**State**

18. Go back to "**Edit**" Action, by clicking on it in Studio left nav.
19. Click on **Action parameters mapping** related List.

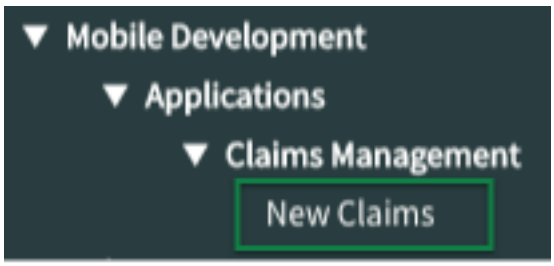
We need 2 mappings

Descriptio	
Item parameter	**description**
UI Parameter	**Description**

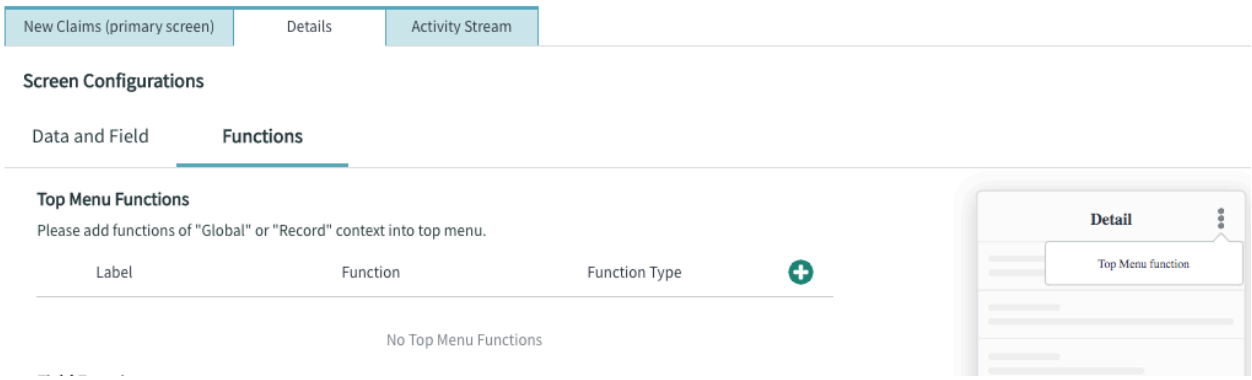
State	
Item parameter	**state**
UI Parameter	**State**



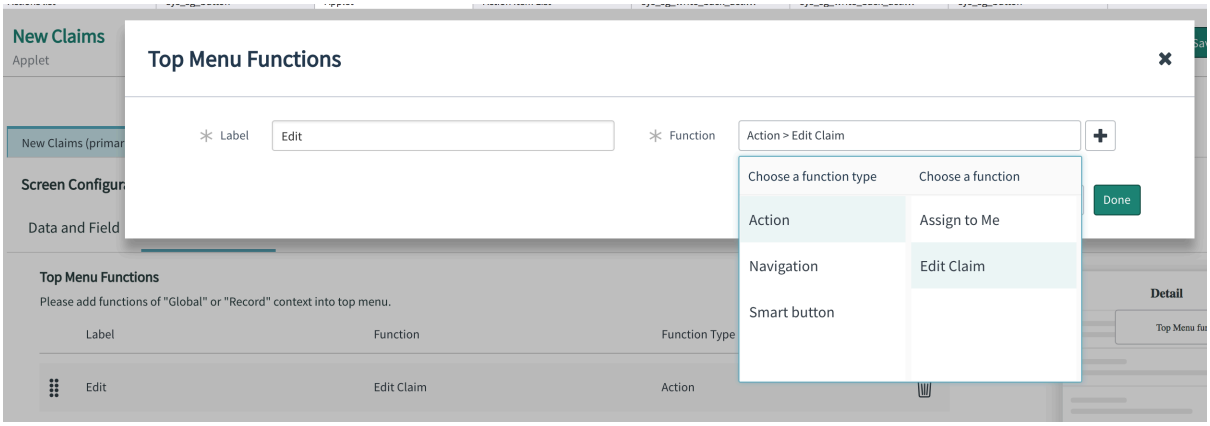
20. **Save** the records
21. Now we need to assign this button to UI
22. Launch our applet "**New Claims**"



23. Click on Details tab -> Functions Tab



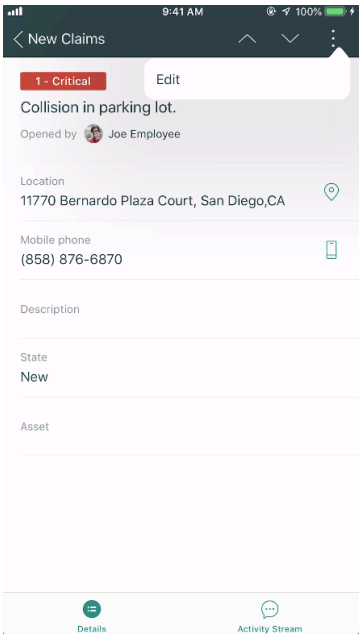
24. We will add this as a top menu function. Click on **+** icon under top menu functions

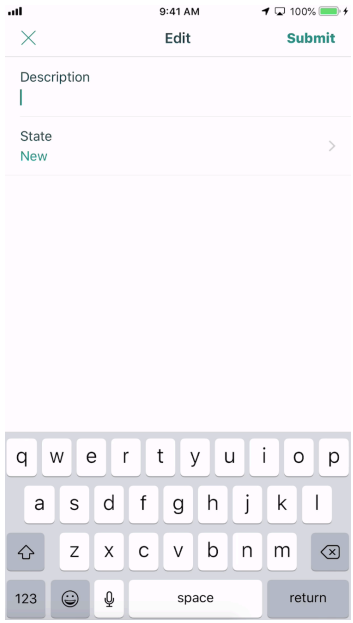


Save the applet!

TIP: You can also add the "Assign to me" we created here to the Top menu.

Verify: The edit functionality works





Lab 6 – Offline

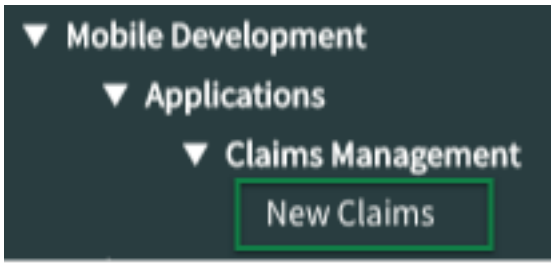
Offline Mode allows users who have no Internet connection to continue working from a mobile device.

For offline to work you must have plugin installed SG Offline support (com.glide.sg.offline). This was done in the first lab. You can configure following to be offline

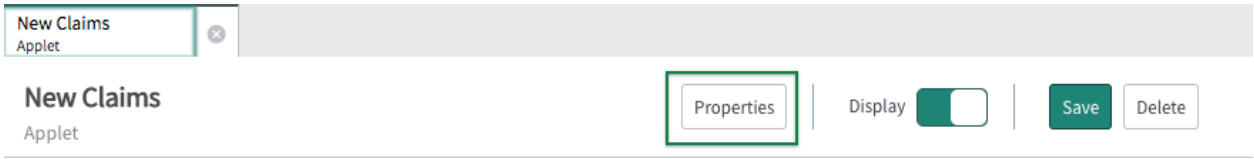
- **Application**
- **Applets**
- **Functions**

Set up our New Claims app to work in offline mode.

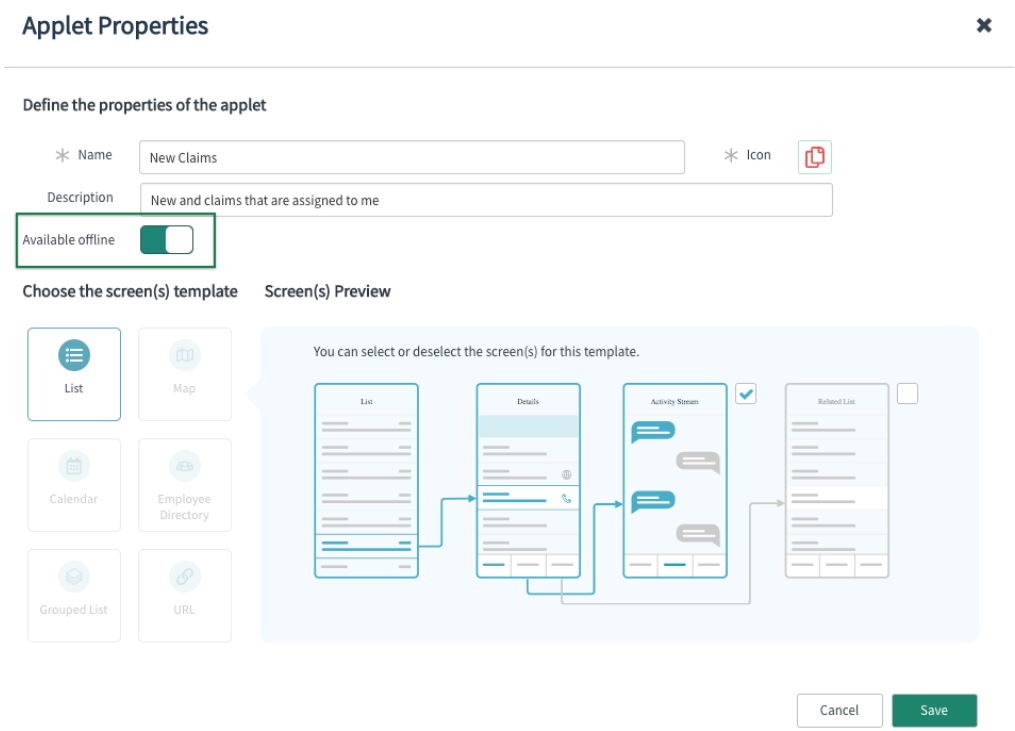
25. Launch our applet "**New Claims**"



26. Click on **properties** button

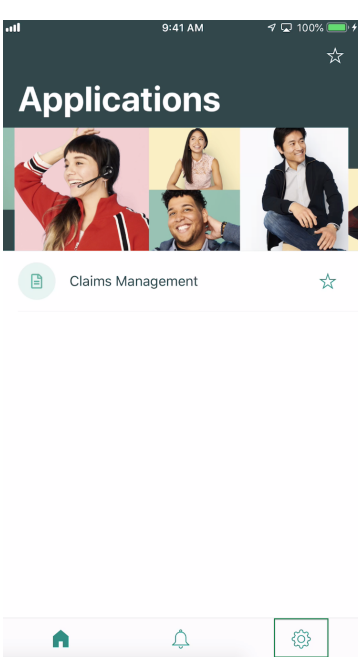


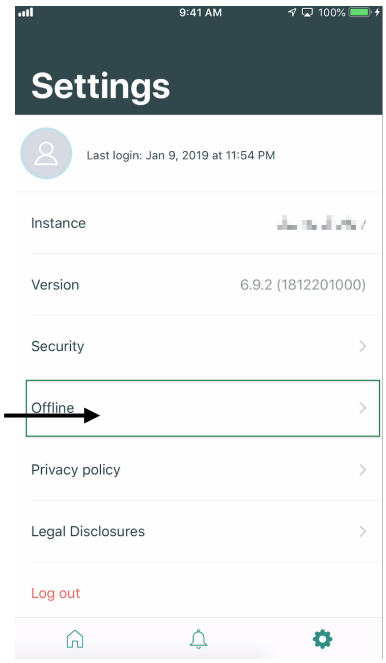
27. Toggle **Available offline** to true and **Save**



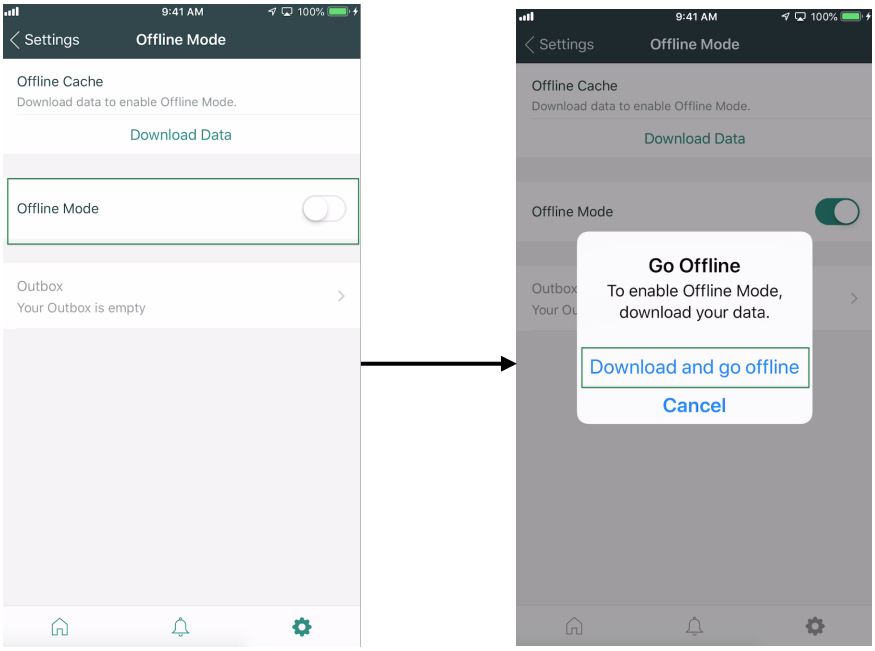
TIP: At any point, if you want to disable/enable Activity Stream, Related list or Change the template to something else you can do it in the properties.

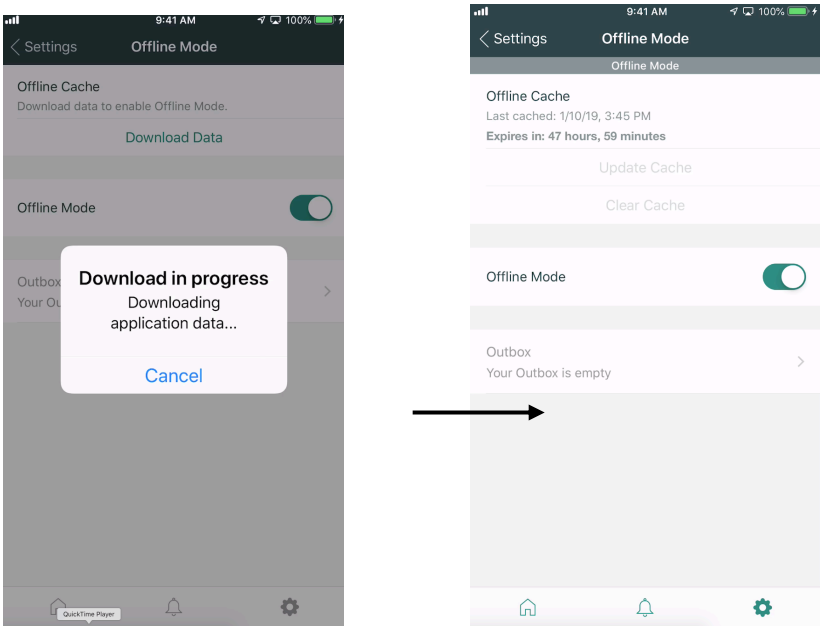
Verify: Toggle the app to offline mode and to test it out.
Refresh your app and follow these steps to enable offline on app



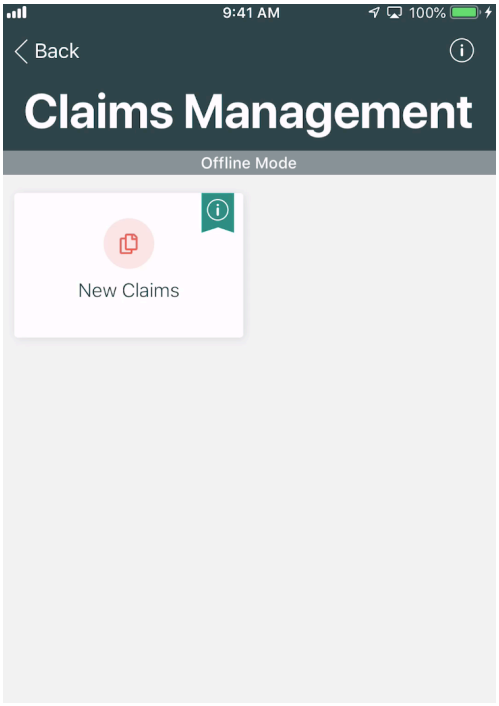


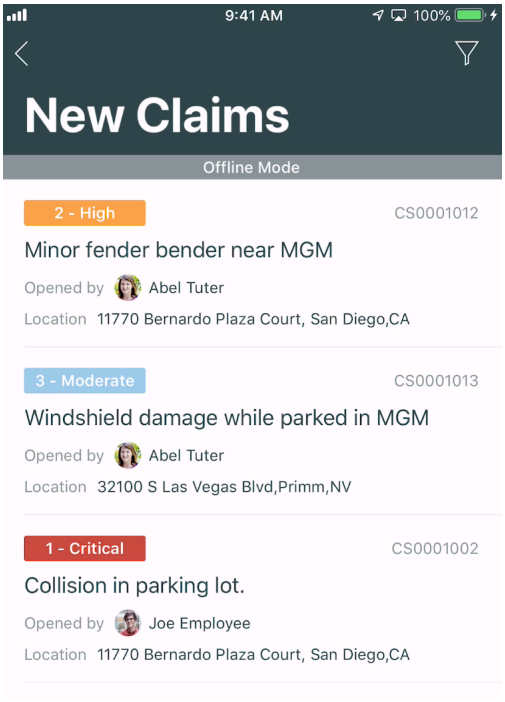
Toggling the "Offline mode" will start downloading the data to phone.





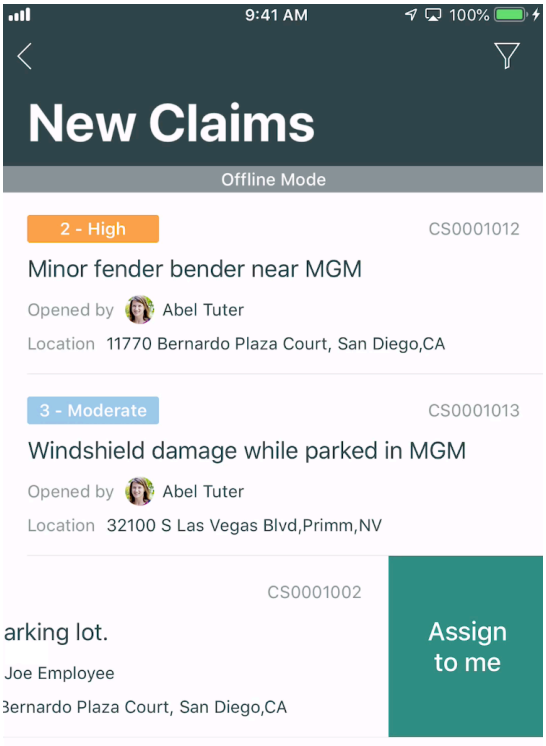
Now the app is in offline mode, go back home and try using the app.



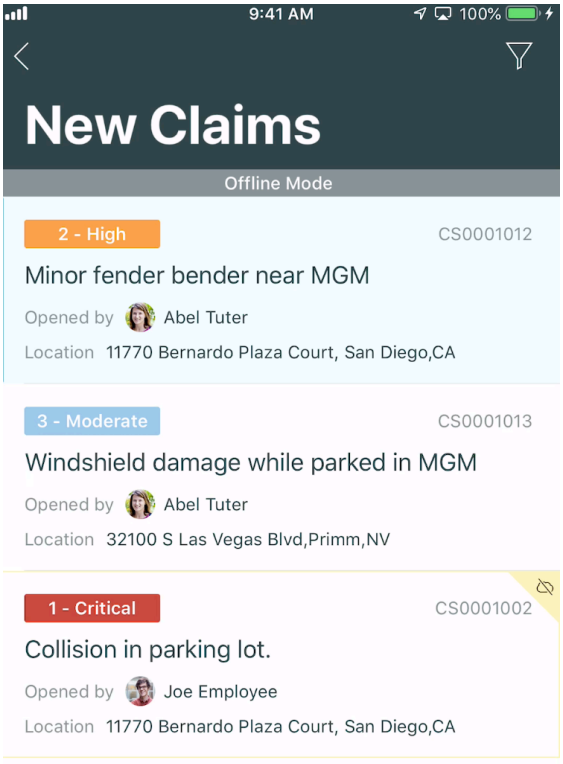


Maps swipe action is not available, we disabled offline for this function in Lab 4 Step 3.

Assign to me is available, we enabled offline for this function in Lab 5 Step 13.

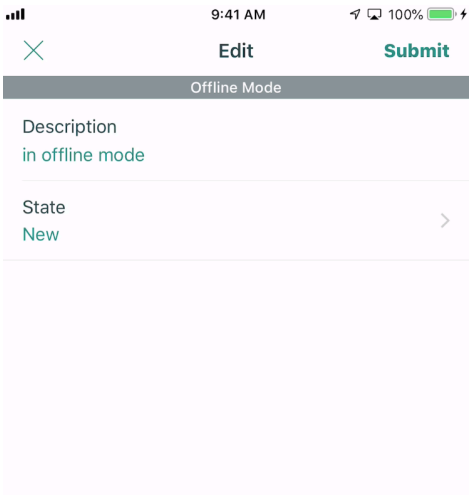


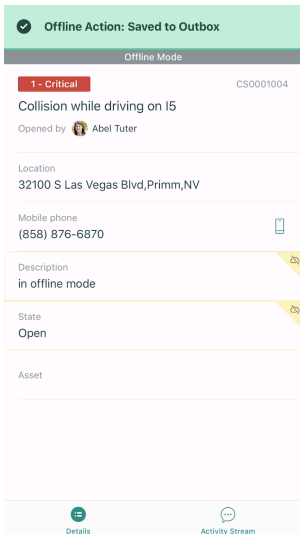
If you use the action "Assign to me", the record gets updated and gives you an indication "it's not connected to internet".



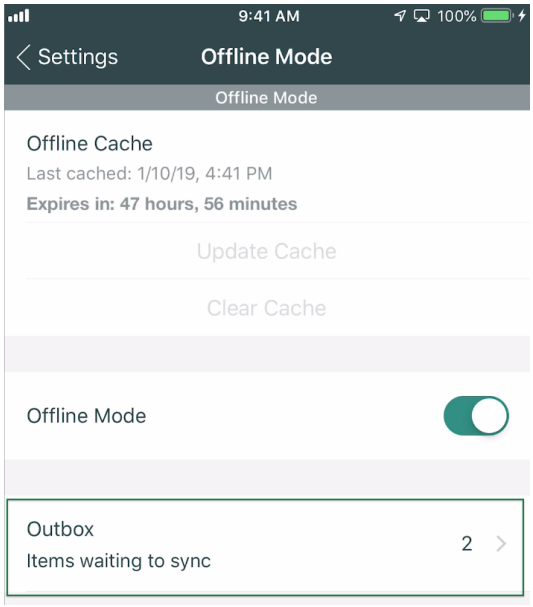
On details screen, Call/SMS function is available in lab 4 and Maps functionality is disabled.

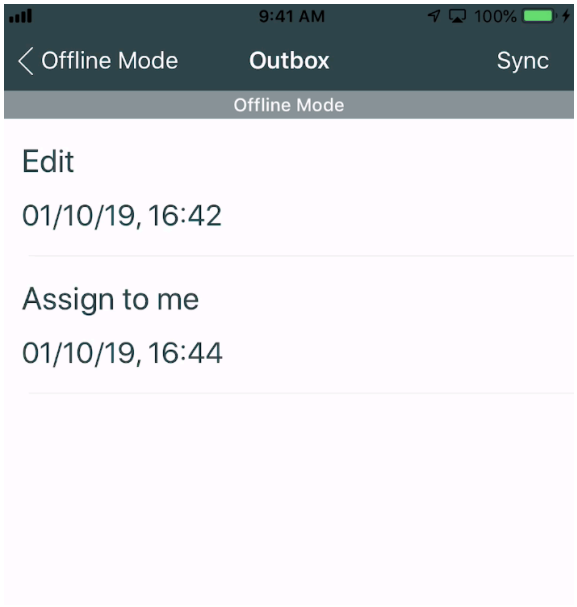
Edit functionality is enabled, you can edit the record. All the updates will be recorded and will be updated once we go online.





Now if we go back to app Settings -> Offline. You will see Outbox. This has the timestamps and the updates that will be applied once we go online.





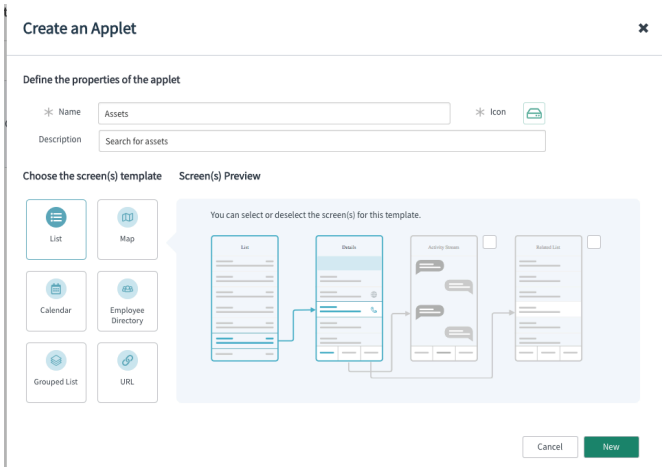
Let's toggle back to online mode. Claim record on the instance should be updated.

Bonus Lab – QR Code Scanner

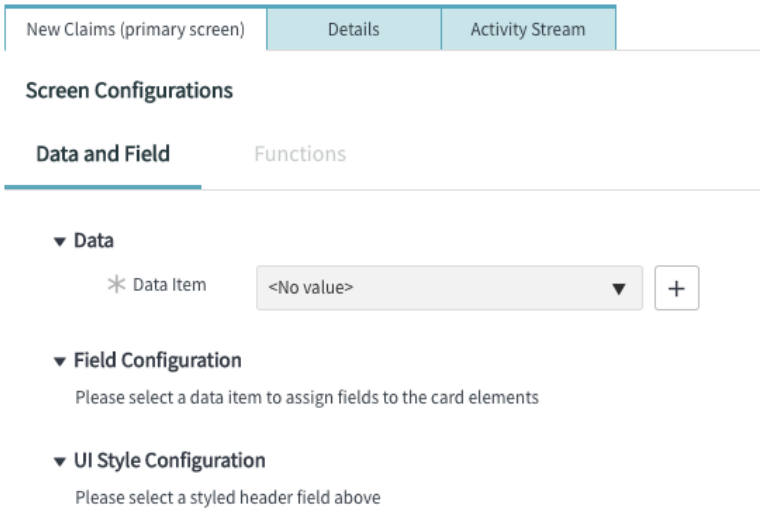
Create a new applet to scan QR Code and Search for Assets

1. **Create a new applet** under **Claims Management** application, by clicking on **pop-out**

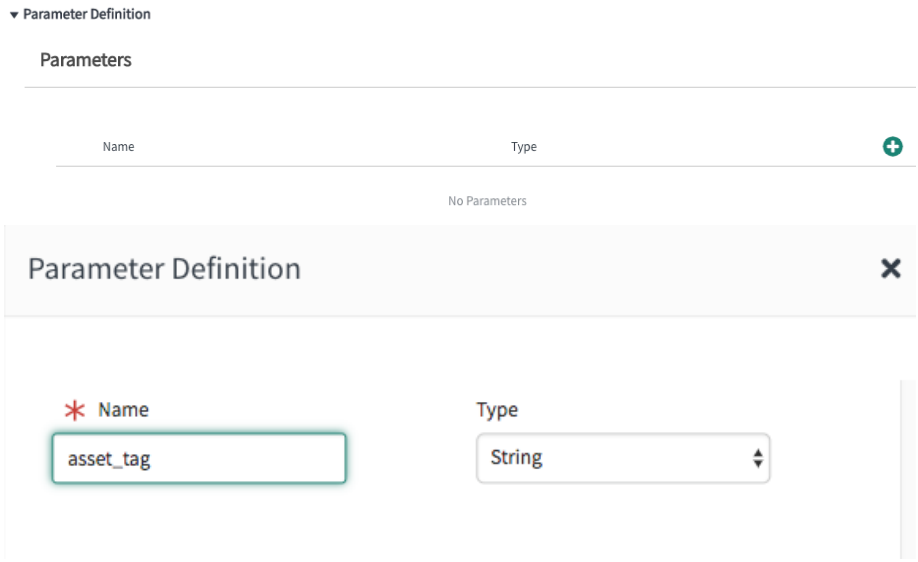
2. Call the new applet **Assets,** choose Icon and color
3. Template type is **List** and click **New**



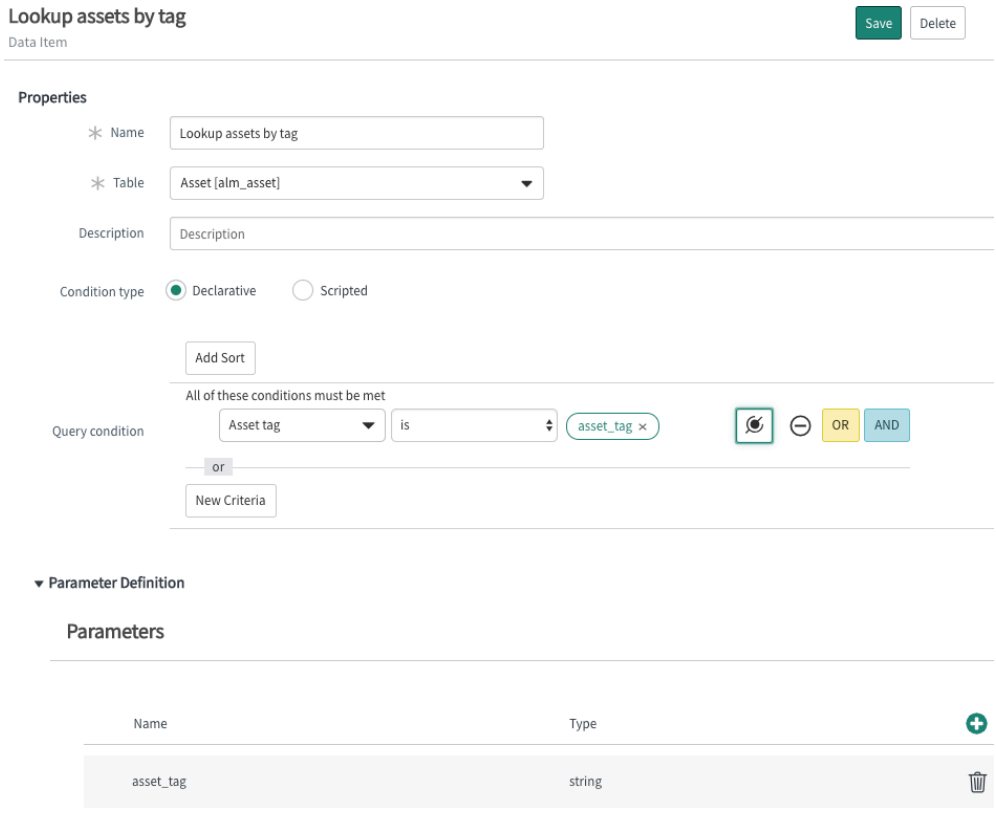
4. Need to create a **Data Item** for the primary screen by clicking on + Icon



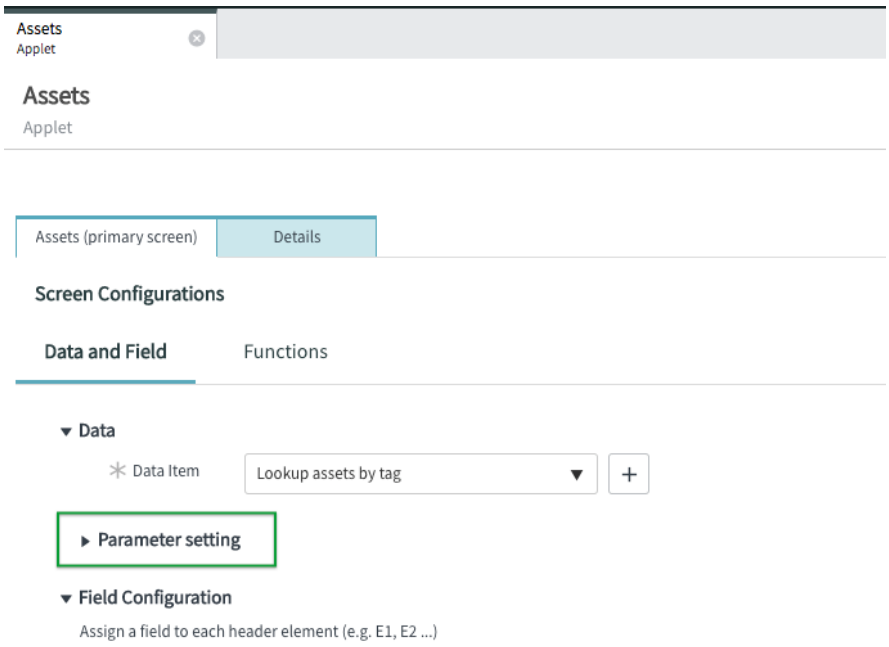
5. Name: **Lookup assets by tag**
6. Table: Asset (**alm_asset**)
7. Save the record
   We need to create a **UI Parameter** by clicking on + icon,



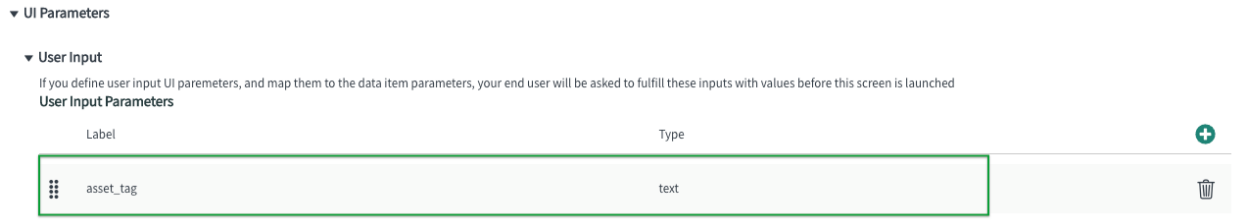
8. Save the record
9. Assign this item parameter to the **Query condition**
10.



11. Save the **Data Item**
12. Now we must set up the **UI parameter "asset_tag" to read a QR code.**
13. Launch the **Assets** applet page.
14. Expand the **Parameter Setting** by clicking on it



15. Open "**asset_tag**" UI parameter record (this was automatically created when we created **Item Parameter**)

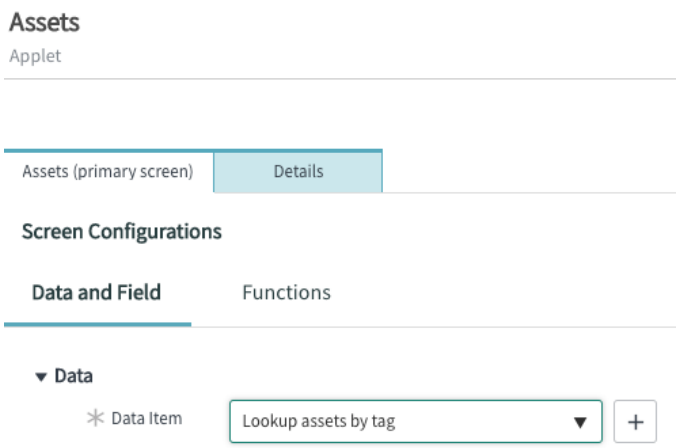


Edit the record with below details and save it.

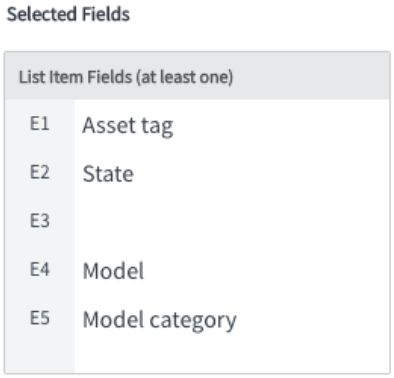
Name	Asset tag
Input type	QR/Code

Save the applet record

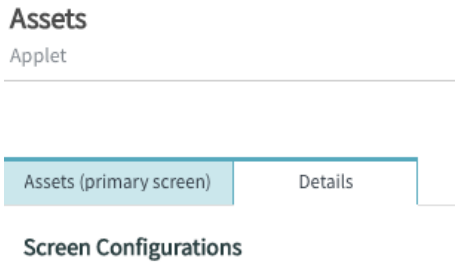
16. Assign this **Data Item** to the Assets applet



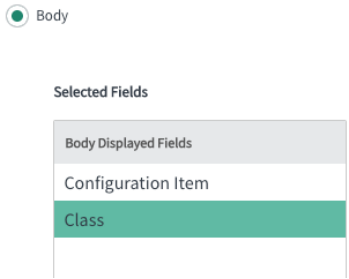
17. Configure the fields like below and **save the Applet**



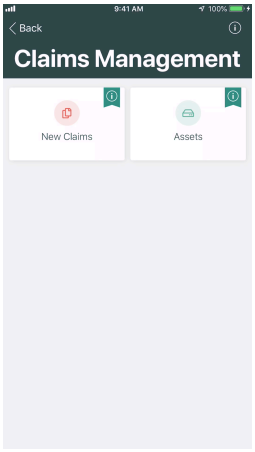
18. Configure the **Details Header**, by **Replicating from primary**

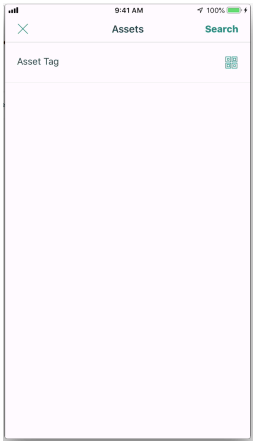


19. Configure body with these fields and **Save the applet**

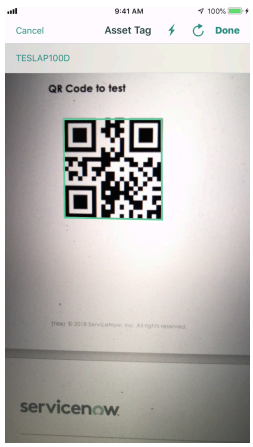


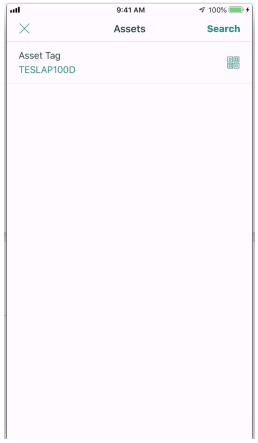
Verify: Open Assets Applet and scan a QR code and search for the Asset.

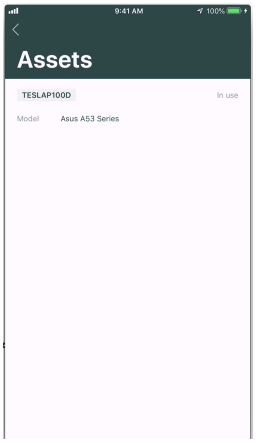


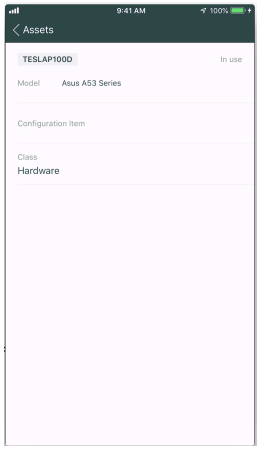


QR Code to test, Once you have scanned QR code, hit the search button









Tip: Import this app to be able to generate your own QRs
https://github.com/arnoudkooi/sn-qrcode.git