Pizza Order Online Requirements

Requirements:

notes: we refer to menu as in restaurant menu (not the application menu)

1) SPLASH IMAGE - Splash image is displayed as app is loading.
   After app loaded:
2) LOAD DATA FROM LOCAL STORAGE - get data from local storage (menu_version_id, last_menu_refresh, menu_data(JSON))
3) CHECK IF MENU REFRESH IS NEEDED - check if is has been more then 2 days that the menu was refreshed (check against last_menu_refresh), if so, menu refresh is needed.
   If menu refresh is needed and menu_version_id=0, then call MENU REFRESH.
   If menu refresh is needed and menu_version_id>0, then call GET MENU VERSION ID (to see if a full menu refresh is needed)
  If last_menu_refresh is less than 2 days ago, then MENU REFRESH is not needed; menu items will load from local storage.
4) GET MENU VERSION ID - HTTP CALL - http call to remote server get the latest menu_version_id.
5) CHECK MENU VERSION ID - check if menu_version_id that came as a result of previous http call is greater than the one from local storage; if it is, then the menu has been updated and a MENU REFRESH is necessary.
6) MENU REFRESH - HTTP CALL - Make a non-blocking/asynchronous http call to a backend restful server that will return json data (of menu items) - display the same splash image with a "loading menu items" message
7) SAVE TO LOCAL STORAGE - save the JSON response that contain the menu items to local storage. Also save a new menu_version_id, last_menu_refresh.
8) render CATEGORIES PAGE - UI LIST VIEW - now that we have the menu items, display the first page which is a list view. Each item is a category (like Salads, Pizza, Desserts . . .). The items have a photo, a name, an id. View mockups.
   User will click on a category, then:
9) render MENU ITEMS PAGE - UI LIST VIEW - another list view, now with all the products for the chosen category. User should be able to go back (pressing the button on the action bar), or click in one of the products. Each item has a number, a name, a small description, a price and an arrow so the user can click to “choose” the item.
   after user taps and chooses one item
10) render ITEM DETAIL FORM - UI FORM - this page has more info on the item and some form buttons; a + and - so user can update the quantity, a checkbox for options, a textField (or textArea?) so that user can enter a special request for that item. The action bar has the usual GO BACK button, which brings the user back to the list (basic table view functionality). The form has a button on the bottom (ADD TO ORDER) so that this item with all the form selections is added to the order.
   10 - A) some items have a flag that allows to be combined with another item, (like a pizza half this flavor, half another flavor); if this flag is set to true (allow_half_flavor==true), then the page has another button “Choose Second Flavor”.

When the button is clicked, the user should go back to the list of items and pick his second item. Then he's back to the ITEM DETAIL FORM, but now it shows the 2 items (flavors) he picked.
11) ADD ITEM TO ORDER - when user clicks the button to add to order, we should add the item to an order object, and redirect the user back to the list of items he was before. ALSO, once there's something in the order object (an item has already been added to order), the ActionBar should have one more button, on the right, for the user to "checkout".
12) render ORDER REVIEW PAGE - UI LIST - when user clicks on checkout button, render the Order page. The order review page has a list with all the items the user is buying, with prices, and the total. A NEXT button so user can continue with his order, or BACK, so he can change it.
13) render ORDER PERSONAL INFO PAGE - UI FORM - now we need to collect some info from the user, in order to send him his order. Name, address, zip code, phone and email. Also, a button "PLACE MY ORDER"
14) SUBMIT ORDER - make an HTTP POST to send all the info collected in the order taking process to backend remote sever. Pack as JSON and send it. The successful response will contain an order number. Clear the Order Object. Redirect user to a new confirmation page.
15) render CONFIRMATION PAGE - UI - last page, needs to say "Thank you, your order has been submitted. Your order number is 222222 (the order_number that came in the SUBMIT ORDER http call response). A button on the bottom of the page "Back to MENU" which should take user to the first page, the CATEGORIES PAGE. His order should be empty.

Mockups of the pages with the design will be provided.