**VoiceList Android OS app**

1. **Introduction and Motivation**

We have developed an android app, which facilitates in generation and collection of a list of items. The motivation behind the app is:

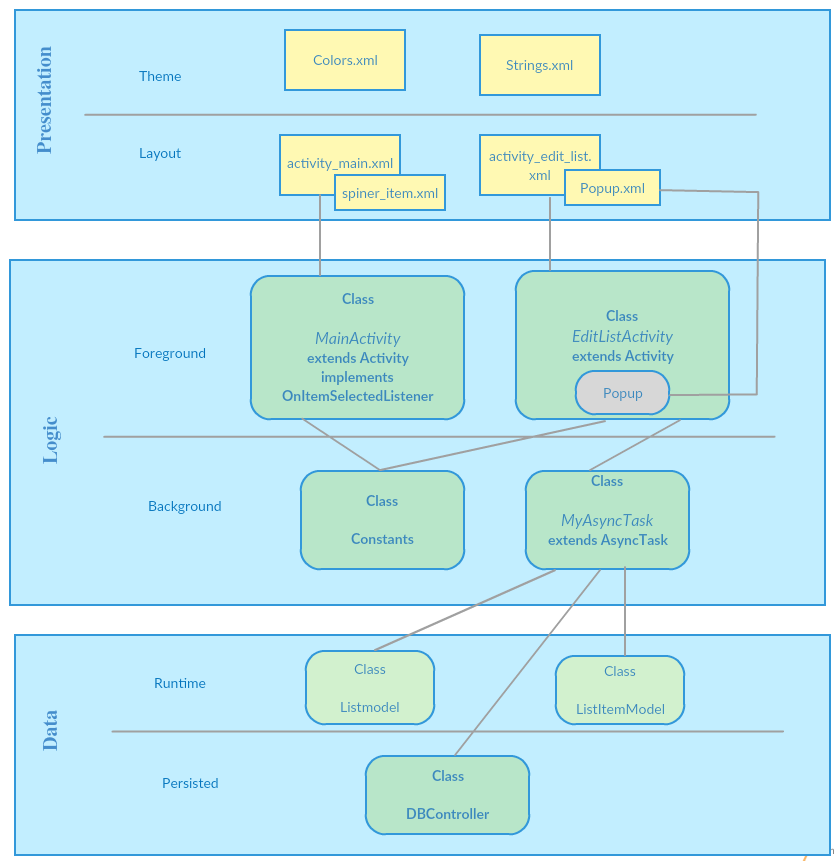
* To provide an easy user interface which enables end user to create the list of items she wants to buy (ex: grocery, utility list, stationary list) using her voice input.
* The user can open the app and add items to a list, whenever she remembers about an item to be bought and doesn’t need to type anything.

1. **Implementation**

Few important features of the application are:

* Provide user an interface to create new list using speech recognition
* Provides user an interface to save a new list or an updated list to the database.
* Provides the user a scroll view to see all previously created list
* User can see sort the list present in the scroll view using different criteria
* User can search for a particular list using filter button
* Enabling and disabling of “Save” button to save the list as per the user’s activity.

1. **Architecture**



**Fig 1: Voice List App Architecture Diagram**

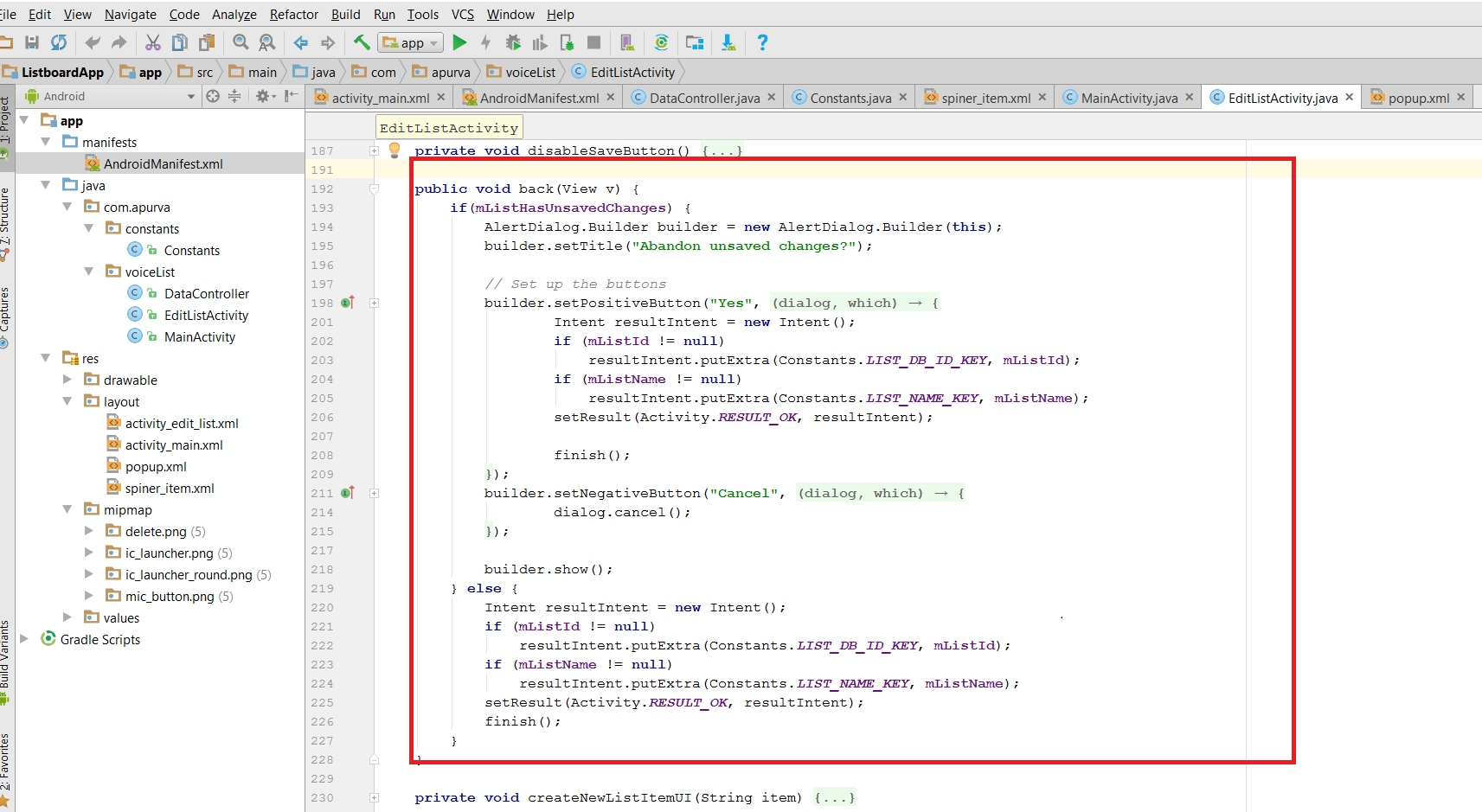
1. **Design Patterns**

This android app follows several design patterns which can be categorized as follows:

* **Creational patterns:** how you create objects.
* **Structural patterns:** how you compose objects.
* **Behavioral patterns:** how you coordinate object interactions.

1. **Creational Pattern:**

* **Design pattern used**: Builder
* **How it works:** This pattern separates the construction of a complex object from its representation. In this way, the same construction process can create different representations.
* **Usage:** In Android, the Builder pattern appears when using objects like AlertDailog.Builder as shown in the code snippet below.

****

**Fig: Builder Design pattern used for dialog screen which open on clicking save button**

1. **Structural Pattern:**

* **Design pattern used**: Adapter
* **How it works:** This pattern lets two incompatible classes work together by converting the interface of a class into another interface the client expects.
* **Usage:** In this app , Adapter pattern is used for the dropdown button present on the app’s landing screen (code snippet shown in the below fig)



**Fig: Adapter design pattern has been used to implement the dropdown buttons list for sorting existing lists using different category**

1. **Behavioral Pattern:**

* **Design pattern used**: Model View Controller
* **How it works:** It refers to the three divisions of classes used in this pattern:
  1. Model: your data classes. If you have User or Product objects, these “model” the real world.
  2. View: your visual classes. Everything the user sees falls under this category.
  3. Controller: the glue between the two. It updates the view, takes user input, and makes changes to the model.

Dividing your code between these three categories makes our code decoupled and reusable.

* **Usage:** In this app , MVC pattern is followed as:

**Model:** listmodel. Java lititemmodel.java

**View:** MainActivity.java, EditListActivity.java

**Controller:** DataController.java

1. **ReadMe**
2. Download/Transfer the .apk file of the project to your device
3. Install the application on the device.
4. Open the application, on the landing page click on “New List” button to create a new list.
5. On the “Create List” screen , click on “Untitled” button present at the top of the screen to enter the name of the new list in dialog box and click “OK”.
6. Click on the “Mic button” present at the bottom of the “Create List” screen to start the Google voice listener and speak loudly the name of the item which you want to enter in the newly created list.
7. The disables “SAVE” button will be enabled as you add any new item to the list. Click on it to save the list to the database.
8. Click on “Back” button to go back the app’s landing screen.
9. The newly created list will appear now in the scroll view present in the middle of the app’s landing screen.
10. Click on the dropdown button and choose a category “Name”, “Created On” or “Updated on” to sort the multiple lists present in the scrollview.
11. Use the “Filter” button present on the app’s landing screen to search a particular list out of multiple lists present in the scroll view.
12. To close the app, click on “Close” button present on the upper right corner of the app’s landing screen

For further assistance, follow the screenshots present in section 6 of this document.

1. **Screenshots:**

|  |  |
| --- | --- |
| C:\Users\Apurva\Dropbox\Screenshots\Screenshot 2017-04-26 14.25.28.png | C:\Users\Apurva\Dropbox\Screenshots\Screenshot 2017-04-26 14.25.38.png |
| Fig 1:Voide List Android app landing screen | Fig 2: User Click on “New List” Button and “EditListActivity” screen is opened. |
| C:\Users\Apurva\Dropbox\Screenshots\Screenshot 2017-04-26 14.25.49.png | C:\Users\Apurva\Dropbox\Screenshots\Screenshot 2017-04-26 14.25.57.png |
| Fig 3: User Click on “Untitled” button to enter the name of the new list in the popup dialog | Fig 4: User enter the name of the new list in the dialog and click on “OK” |

|  |  |
| --- | --- |
| C:\Users\Apurva\Dropbox\Screenshots\Screenshot 2017-04-26 14.26.18.png | C:\Users\Apurva\Dropbox\Screenshots\Screenshot 2017-04-26 14.28.03.png |
| Fig 5: User clicks on Mic button to start the goggle voice listener. | Fig 6: User speaks to enter two items in the list and clicks on “SAVE” button |
| C:\Users\Apurva\Dropbox\Screenshots\Screenshot 2017-04-26 14.28.24.png |  |
| Fig 7: The newly created list can be seen in the scroll view. | Fig 8: User clicks on sort by dropdown menu and selects ‘Name’ category to sort the lists |
| Screenshot |  |
| Fig 9: User clicks on sort by dropdown menu and selects ‘Created date’ category to sort the lists | Fig 10: User clicks on sort by dropdown menu and selects ‘Updated Date’ category to sort the lists |
|  |  |