1. Create a new colour resource to be used as the background for the data entry part of the MainActivity. Search on the web for the color and associated Android color code (color codes always start with a # symbol), and add it to the color.xml file. Set the background of the data entry part of the layout to that color resource.
2. Make the navigation work for all activities in the app. Copy the navigation bar XML code to the layout associated with each activity. Copy the Java code that makes the buttons work to the Java file associated with each activity. You will have to modify that code to reference the activity it is in, rather than MainActivity. Add code to disable the ImageButton associated with the activity that is displayed.
3. Modify the DatePickerDialog layout so that the Cancel/OK buttons are centered. Hint: You’ll have to use the gravity attribute in the LinearLayout.
4. Graphical user interface, application

   Description automatically generatedCreate project called ContractorCalculator.
5. The main activity layout should contain two EditTexts, one button, and eight TextViews
6. The Calculate button should add the labor and material costs from the EditTexts and put the result in the TextView next to SubTotal. Tax should be calculated using a 5% rate (use a constant) and displayed in the TextView next to Tax. The tax and subtotal should be added together and displayed in the TextView next to Total.
7. Create a project called MealRater.
   * A picture containing graphical user interface

     Description automatically generatedThe activity layout should contain two EditTexts, one button, and four TextViews
   * One edit text will allow entering the name of the dish and the other the name of the restaurant.
   * The button will open a dialog that will allow the user to rate the meal on a 1 to 5 scale and then display that rating in a TextView.

•

1. Modify the Contact List app created in **Chapter Navigation AND Interface Design** to do the following:
   * Create a custom dialog to ask users their relationship to the contact. Use radio buttons to present the following choices: Friend, Family, Coworker, and Acquaintance. Add a TextView to display the relationship below the contact’s name. Add a button to open the dialog before the relationship TextView. The custom dialog should return the selection when the user taps the OK button on the dialog. It should display Not Set if the user taps the Cancel button.

1. Add the choice of a background colour to the settings activity. Create a couple of new color resources in color.xml. Add these choices as a RadioGroup to the settings screen. You will have to modify the layout to place all the RadioGroups in a ScrollView so that you can see them all. Make the choice persist in a SharedPreferences object. Use the following command in the onCreate method of the settings activity to set the chosen background color:

***scrollviewobject.setBackgroundResource(R.color.colorresourcename);***

1. Create a method in ContactDataSource that will only update the Contact’s address. Create a ContactAddress object to pass data to the method.
2. Modify the Contact table to include a field bestFriendForever that is an integer data type. Modify the onUpgrade method of ContactDBHelper to insert this new field without losing the data that is currently in the table.
3. Modify ContractorCalculator to display the tax rate. Use two TextViews: One to display the text “Tax Rate:” and the other to display the current tax rate. Add a button, “Change Rate,” that will open a custom dialog that allows the user to enter the tax rate. Save the tax rate in a SharedPreferences object. When the dialog is closed, the new tax rate should be displayed in the TextView. The app should use this rate for calculations until it is changed. The tax rate should be loaded from the SharedPreferences object anytime the app opens.
4. Modify the MealRater app created in Question 1, to include a Save button. The button should save the restaurant name, meal, and rating to a database table. Create a DBMealRaterHelper class to create the database and a MealRaterDataSource class to provide the SQL insert statement for the data.
5. Create an app called MyFinances. The app should allow the storing of financial objects. The financial objects are CDs, Loans, and Checking accounts. CDs should be able to store the account number, initial balance, current balance, and interest rate. Loans should be able to store the account number, initial balance, current balance, payment amount, and interest rate. Checking accounts should be able to store the account number and current balance.
   * The user should select the type of account to enter with RadioButtons.
   * Selection of the account should enable/disable relevant fields on the data entry activity.
   * Store objects using a database table when the user clicks the Save button.
   * The Save button should also display a saved message and clear the screen.
   * A Cancel button should clear the screen without saving any data.
6. Create an app called HotSpots! The MainActivity should allow the user to save the name and address of a bar or nightclub into a database table. Add a Rate button to the activity. The Rate button will open a custom dialog that will allow the user to rate the establishment on a 1 to 5 scale for each of three dimensions: beer selection, wine selection, and music. Save the ratings in the table and display the average rating on the MainActivity when the dialog is closed. Each rating should be stored individually (a separate field) so they can be adjusted later.
7. Create an app called Super!Market. The MainActivity should allow the user to save the name and address of a supermarket into a database table. Add a Rate button to the activity. The Rate button will open a second activity that will allow the user to rate the establishment on a 1 to 5 scale for each of five dimensions: liquor department, produce department, meat department, cheese selection, and ease of checkout. A Save button will store the ratings in a table and display the average rating on the Rating Activity. Each rating should be stored individually (a separate field) so they can be adjusted later. Add a button to navigate back to the MainActivity.
8. Create an app called RestaurantRater that allows the user to save the name and address of a restaurant into a database table and to save entrée, appetizers, and desserts associated with a

specific restaurant in a second related table. The second table should contain the following fields: name (of dish), type (entrée, dessert, etc.) and a rating on a 1 to 5 scale. The MainActivity should allow the saving of the restaurant and its address and include a button to open a RateDish activity. This button should pass the name of the restaurant (or id) to the activity. This value will be used as the foreign key in the second table.

Diagram

Description automatically generated