Graphic User Interface:

This utilized Android Studio’s graphical interface function. These are all turned into XMLs which are located in the layout folder. Each page has a content.xml and activity.xml. The content files were edited to include a common background, buttons, images, and align all the aforementioned with the necessary text fields and text boxes. Drawables were imported on Android Studio for the title and image on the homepage, as well as header image and image buttons on the pages that followed.

Backend:

For this part we utilized Java scripts, giving each page its own Java file to call the appropriate functions. At the top of each page we import the necessary library for those functions.

Main Activity:

We used an OnCreate function to setup the page and access the XML files. This page also contained a button which, when clicked, linked the user to the home page.

Home Page:

We used another OnCreate function to create the page and access its XML file. Next, the user is prompted to enter the Group Name as well as the number of guests. These inputs are stored, and using an intent we made the variables accessible across all activities.

Contact Page:

The number of guests the user entered is then used to create the number of Contact Pages that will be displayed to the user. On each Contact Page, the user is asked to enter the person’s name and email. This information is stored in two arrays, one containing all emails and one containing all names. The indices of each array correspond to ensure that the pairs are kept together.

Meeting Page:

This page prompts the user to enter a meeting location, time, and date. Again, these values are shared through the intent functionality and utilized in the final page.

Final Page:

We created a function to randomize the order of the names and emails in the data structure, ensuring that names and emails were kept paired. On this page we utilized a JavaMail API that is discussed below. We created another function that called the Send Mail class, and sent each person the name of the person following them in the array. The final person in the array received the name of the first person. This ensures that there are no repeats, pairs, and no one gets themselves. In addition, this page displays the group name, the meeting location, date, and time, all of which are also included in the email. Finally, the page has two buttons. The “Start Over” button will allow the user to return to the home page and clear all entered information. The “Send Email” button sends the emails to each guest.

JavaMail API:

We used simplifiedcoding.net as our reference. In order to do this, we needed to take three .jar files – name1, name2, name 3 – and added them to our source library. Next, we went into the project structure and made file dependencies. Next, we created a gmail account to send the emails from. We made a config.java file where we declared this email address and password. We made a class called sendmail.java which we took from Simplified Coding and edited for our purposes. Finally, we changed the manifest to allow permission to access the Internet and send out the email.

Start Over

Send Emails

Final Page

Meeting Page

Contact Page

Home Page

Main Activity