System Manual
PartyUp App
Group #1

Importing:
Each .swift file has its own imports at the top of the code. These imports allow certain functions to be used which might not be readily available otherwise. Import Firebase, for example, will import all of firebases functions which allow the programmer to pull and write data from the database.

Register Screen:
The purpose of this screen is to allow new users to create their new accounts if they have never visited the app before. This screen has two text fields, a submit button, and a button that will take you to the login screen if you already have an account. When the register button is clicked, the data that is inside of the text field will be taken and entered into the Firebase database where they will have an account created for them. This will also take their Drexel University ID and store it as a global variable so that it can be used when they are creating a party. If there was an error, they user will be prompted to fix that error and try again. Examples of errors could a be username that has already been created or a password that does not fit the criteria. If everything is okay and there are no errors, the user will then be taken to the account info screen.

Account Info Screen:
This screen will allow users to make their profile more detailed and also give the user a chance to accept the terms of service. The user will start by entering their name into the corresponding text fields and flip the switch agreeing to the PartyUp TOS. Once the user agrees, a button will pop up allowing them to move onto the next screen. If the user presses the button and any of the text fields (first or last name) are blank an error message will pop up and they will need to fill out any missing fields. In the event that nothing goes wrong, clicking the button will save their first and last name under their Drexel ID in Firebase and move onto the login screen.

Login Screen:
The login screen allows users who have already created accounts to access the mobile applications features. The user will need to enter their Drexel ID and password into the corresponding text fields and press login. When the user clicks the button, the app will check to Firebase to make sure that the email and password

are both correct so that the user can continue on. If the information is false, the app will go through the procedure of displaying a window of what is wrong. If the information is correct, the username will be saved into a global variable for later usage and the screen will segue to the Tab Bar controller.

Forgot Password Screen:
This screen will allow the user to reset their password in the event that they forgot it. It does this by taking the email that is entered in the text field and running it through firebases pre-created forgot a password function. This function simply takes the email that the user entered and send an email to that person allowing them to change their password.

Tab Bar Controller:
The purpose of this controller was to allow the user to quickly move between the three screen that are attached to the controller. Tab bar controllers allow for multiple screen to be accessed with the swipe of the screen. The three screen involved in this controller are the search screen, the host screen, and the settings screen. Each of these screens have a different picture at the bottom so that the user can tell which screen they are on.

Navigation Controller:
The navigation controller allows the user to hit a back button at the top left of their screen. This button will segue back to the previous screen and keep all the data intact. This is vital for moving around the app without having to create segues all over the place.

AdMob:
The main source of income for this app is through the ad revenue. Each of the main pages of the app have individual ad banners at the bottom which will generate revenue when people see them. Each ad has its own banner id that connects to the AdMob servers. For the purposes of testing, the ads are in simulation mode because the app is not yet on the App Store. If it were on the store, the simulation mode would have to be removed by entering in the true ID instead of kGADSimulatorID.

Search Screen:
This screen allows guests to find parties that were created and stored into Firebase. They are allowed to join the parties and see the people we are attending the party. Each of the currently available parties are displayed in the table view with each party have its own cell. The table is created by making a listener that will

automatically refresh the screen and look to see if any new parties were created. If a new party was created, the next available cell will be where the party is put. Each snapshot of the listener would be one party which is found by the function going through the database and looking under the directory we set. The table view can have infinite scrolling so that if the number of parties goes off the screen, the user can scroll down and see the rest of the parties. The user will be able to click any party they want and quickly be taken to a screen which will show them all the information about the party.

CurrentPartylist Screen:
When the user clicks on a party they want to view, they are taken to this screen. This screen will pull all the data from Firebase that pertains to the individual party and display it for the user to see. Things like the name of the party, the location, the cost, etc. will be pulled from the database directory and placed into the labels. If the user decides they want to join said party, they can click the join button which will automatically pull from their user ID global variable and put their name in the database as someone who will be attending. Their name will also be added to the attendees list at the bottom of the screen which works similarly to the Search Screen's table view. It will create a listener in the attendees' directory and update it if any changes are made. If a user wants to leave a party, they can click the leave party button at the top right of the screen. This will remove their name from the directory in firebase and update it on the screen.

Hosted Screen:
This party is almost exactly the same as the Search Screen with one minor adjustment. Rather than display all the parties that are in the database, it will only display the ones that were created by that individual user. This is done by assigning each party with the global variable user ID of the user who created it which will prevent anyone else from making changes. Similar, to the Search Screen, the user can click on any party they created and be taken to a screen where they can make certain changes. If the user wants to create a new party, they can click the Create a Party! button which will segue them to a new screen that allows them to, for lack of better words, create a party.

Create Party Screen:
The purpose of this screen is to take the information that was inputted and create a new party in the firebase database for it. Each of the data fields will need to be entered unless it is not important and all of the data will be used to create a party. When the submit button is clicked, all of the data in the text fields is sent to the database under the directory of the party name. This will keep all of the

information together and easily accessible when needed. If for any reason the party name is already taken, a prompt will appear telling the user to change the name because it is already taken. This screen will also prompt the user for their paypal.me account if they decide to charge money for their party.

Party Info Screen:
This screen allows the host to make changes to a previously created party in the event that it is needed. When the screen first loads, the party directory will be pulled and the information from firebase will be place into the corresponding text fields. The host will then be able to make changes where allowed by simply editing the fields. When the submit button is clicked, the new text fields will be sent to firebase and they will override the old information.

PayPal Screen:
The screen is used to pay the host for the fee of the party. Users can sign into their PayPal and transfer money. The screen does this by using a web view that navigates the user to the host's PayPal.me

Settings Screen:
This screen has multiple maintenance functions which segue to other screens upon being clicked. The logout button will simply take the user back to the main screen and require them to sign back in to access the application. Joined parties, report problems, and feedback will all segue to different screen where the actions will then be carried out. Both of the report problems screen and the feedback screen simply take the line of text and save it into Firebase.

History Screen:
The purpose of this page is to allow people to rate their past parties. The values are in between 1-5 and rounds to the nearest half star. These values will translate over to Firebase and save the values in the parties' child directory. All of the ratings are averaged out and the host is assigned an overall rating. When the user clicks the party they want to view in the table, the party will be segued to and the user can rate the host. Even if the user changes their rating, it will still only count once.