LAB 7

Objectives

1. Identify the entities of a detailed design for your project.
2. Create detailed designs for a subset of the entities of your project.
3. Assess your team's technical capability compared to the technical needs of the project.

Designers need to specify the details of the entities that make up the system. These definitions should be sufficiently detailed that the design can be given to a developer and the developer can create the entity as envisioned by the designer.

Once your team starts to develop a design, you should also be developing a better understanding of the technologies and skill levels needed to build the product. As a separate task, this lab will also provide a chance for you to assess your team's capability to work on the project and identify learning or skill development you may need.

Procedure

Step 1 – Draft a list of entities for your project

You should consider the following types of entities:

- Screens (or Web pages)
- Database tables
- Files (e.g., data that is stored as part of the system but not stored in a database)
- Code (modules, objects, or functions)

Use Figure 7-1 to list all the system entities that you can identify. A good way to start is to pick one area and focus on that. For example, if your system has a significant user interface, start by trying to name all the screens that would comprise your interface. For each entity you list:

- Enter a type, e.g., "screen"
- Give it a meaningful name, e.g., "Customer Profile"
- Provide any short notes or explanation needed to identify the screen, e.g., "This screen captures customer information and preferences."

Step 2 – Create detailed designs for at least 4 of your entities.

You will not be able to design all the entities of your system in this lab, but this step will get you started. Pick 4 entities that you think you understand the best at this point, and create a design for them. Every entity should have a name, type, and design details. Templates are provided to help you create detailed design for screens, database tables, and code functions.

Step 3 – Review your detailed designs.

After creating your designs, review them for completeness and clarity. Ask yourself this question: "If I was the developer and a designer handed me this design, would I know what to build without needing to ask a lot of questions?

If you have created the design entities as a team, set them aside for a few minutes before review each one. If you have worked in sub-groups within your team to create the designs, then exchange designs so the reviewer is a different person than the creator of a design.

Revise your designs based on the review.

Step 3 – Assess your team's capability to complete this project.

Once you have an architectural overview and the beginning of a design, you should be able to assess capability and identify things that someone on the team may need to learn. Use Figure 7-5 to summarize this information.

3.A – List the technologies you need for your project using the column on the left. Consider things such as programming languages, operating systems, specialized data sources, software libraries, support tools, and hardware.

3.B – List each team member at the top of a column, and then evaluate that person's knowledge of the technology in each row. For the column for each team member, use the following values:

1 – No knowledge or not much relative to the needs of this project
2 – Enough knowledge to accomplish part but not all of this project
3 – Knowledge probably sufficient for this project

3.C – Discuss within your team how you will start to gain capabilities that you are missing. You do not need to turn in results of this discussion in this lab, but will need to address this in the coming weeks.

What to Turn In

In order to obtain full credit for this lab, ***each team*** must turn in:

1. Figure 7-1 – Possible System Entities
2. Detailed designs for at least 4 entities in your system. Use the templates in Figures 7-2 through 7-4 to get started.
3. Figure 7-5 – Team Capability Assessment

Figure 7-1 – Possible System Entities

Product: BuddyU
Team: 084
Date: 2/24/2017

Type	Name	Description or Notes
Screen	Login	This will allow the user to log in to the app. ***The user will be directed to this screen if in the registration screen if they hit the textview that they already have an account.***
****Screen* (added 4/7)***	***Registration***	***In this screen, new users will be able to create their accounts; this is the first screen users will see when they open the app for the first time.***
Screen	Home	This will be the screen the user will see after they log into the app. ***It will list all of classes and events for today.***
Screen	Calendar	This will be the user's personal weekly calendar that will show the classes the user has signed up for and the events the user has shown interest in.
Screen	List of Courses	This will be the screen displaying the information pulled from Drexel Term Master Schedule. This will list all the available courses offered from at Drexel University during the quarter.
Screen	Chat/Messaging	This will be the screen the user uses to chat with other friends, or classmates.
****Screen* (added 4/7)***	***Profile Screen***	***This screen will show the email from whatever email the user signed in with. Followed by a profile picture which they can insert and a buddies button. Once the buddies button is clicked, it will show all the buddies the user has and the user will also be able to add buddies in this screen.***
Database Table	Facebook Accounts	This will take in user information from the Facebook API's and store only the necessary information in our customer database.
Database Table	Drexel Accounts	Because there are two tentative options on how the user can signin, this is the second option. We can connect our user account to the Drexel database and put it into our database.
Database Table	Courses Offered	This table will contain all the information about what courses are offered. Pulled from webTMS and stored into our database, from which the user can pull and store into their account.
Files	Chat Memory	This will store messages of chat sessions for future use by users
Files	Event Log	This will keep track of application activity to aid in identifying issues/ bugs that need fixing
Files	Images	Includes application logo, icons, app art, etc.
Code		

Figure 7-2 – Template for Detailed Design for a Screen
Name: ***Registration***
Type: ***Screen***
Purpose: ***This will be the screen the user will arrive at when you first open the app.***
Description: ***Figure 1 shows the layout for this screen. This screen will allow the user to register an account if they already do not have one. The screen contains the following elements:***

A section to enter the the username and the password for an email and a button to register that email.
Two other buttons that allow users to sign in with Google or Facebook accounts.
If the user already has an account they can click the section with: "Already have an account? Login here!" which will direct them to the login screen.

Layout:

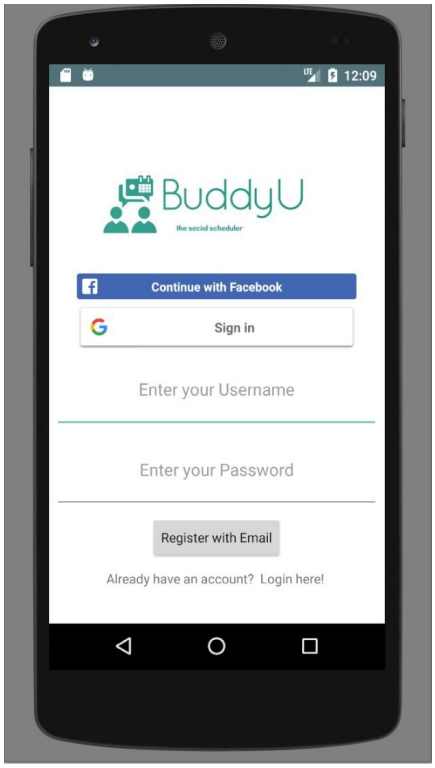

Figure 1 - Registration Screen ***ADDED 4/7***

Figure 2-Login Screen

Name: Login
Type: Screen
Purpose: ***This will be the screen the user will arrive at after you click the text view on the bottom of the registration screen.*** This will allow the user to sign in if they already have an account.
Description: Figure 1 shows the layout for this screen. This screen will allow the user to log in to the app. The screen contains the following elements:

A login section to enter the app containing the username and the password.
A button that allows users to create new accounts using ***email, Facebook, or Google accounts***.

Layout:

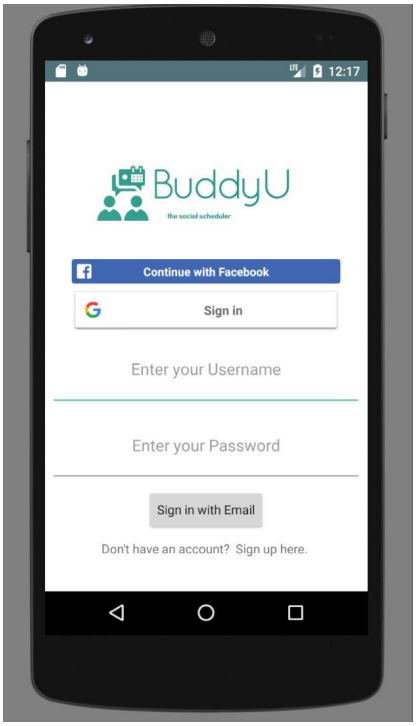

Figure 2 – Login Screen*****UPDATED 4/7*******

Figure 3-Messaging Screen

Name: Messaging
Type: Screen
Purpose: This screen is needed to show a chat feature that we will have if the user clicks on that feature
Description: Once clicked on the chat feature, there will be a screen that lists all the ***individual*** chats and group chats that the user is a part of and once the user ***taps that specific chat***, they will be directed to another screen that will list all the messages in that group/***individual chat that the user selected***.

This screen contains the following elements:
Different chats the user is part of including individual and group chats.
A button that the user can decide to create a new chat.

Layout:

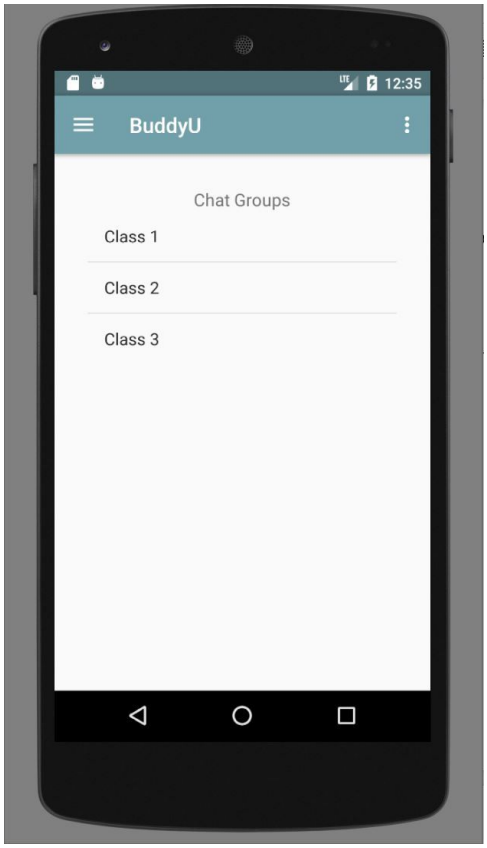

Figure 3 – Messaging Screen ***UPDATED 4/7***

Figure 4-Home Screen

Name: Home screen
Type: Screen
Purpose: This screen will be directed to once the user has finished signing in with their account information.
Description: Figure 4 contains the main bulk of this app. It includes a welcome message ***with all of the current day's classes and events and*** contains a dropdown that lists all our features. These features include current profile of the user, calendar, chat, settings and a Logout button.

The screen contains the following elements. This screen contains a dropdown that contains different features, that when clicked will direct the user to different screens. It will also include a Logout and settings button if the user needs help in any function.
Layout:

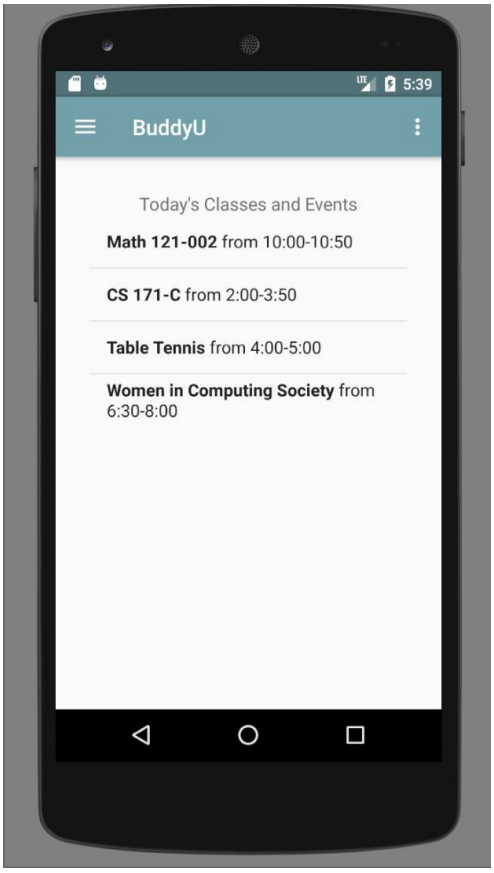

Figure 4 – Home Screen ***UPDATED 4/7***

Figure 5-Calendar Screen

Name: Calendar
Type: Screen
Purpose: This screen is needed to show the user their schedule for the daily and weekly calendar format layouts for the user and their friends.
Description: Figure 5 shows the calendar screen for our app. It will have a daily or weekly format option. This will display all the classes and the available times of the user.

This screen contains ***buttons*** that can change the layout of how the user wants to view their calendar: ***monthly***, daily or weekly. There will also be a button that the user can add an event to their calendar.

Layout:

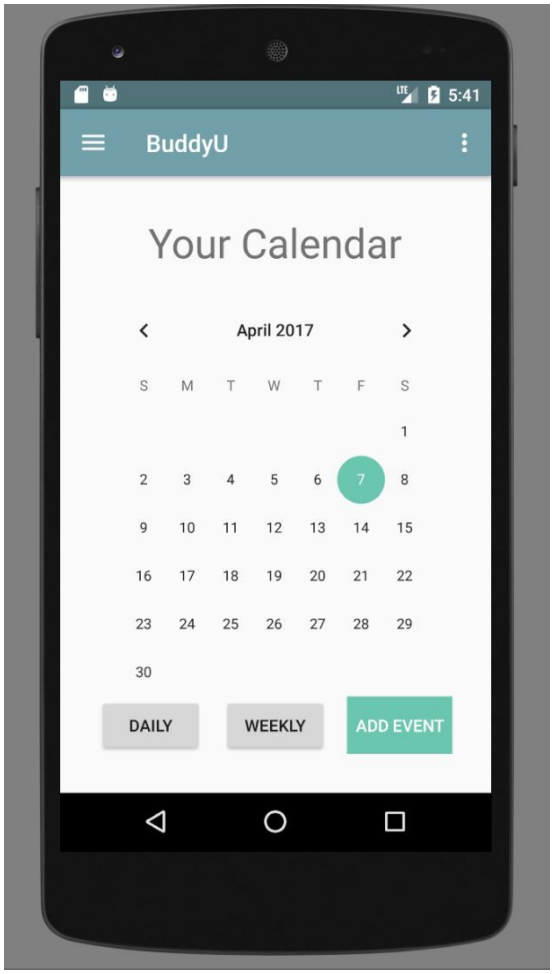

Figure 4 - Calendar Screen ***UPDATED 4/7***

Figure 5-Profile Screen

Name: Profile
Type: Screen
Purpose: This screen is needed to show the user their profile where they can change their image and a buddies button.
Description: Figure 5 shows the profile screen. The profile screen will contain the email the user used to sign in, an image of the user that the user can change if desired, and a buddies button. The buddies button will lead to another screen listing all buddies of the user and will allow users to add buddies.

Layout:

Figure 5 - Profile Screen ***ADDED 4/7****

Figure 7-3 – Template for Detailed Design for a Database Table
Name: BuddyU Database Table
Type: Database Table
Purpose: This table is needed to figure out and finalize what content our database using MongoDB will be holding.
Description: Figure 5 shows the contents for this table. This table shows the different data elements we will need for our app, BuddyU, and describes the data type for that certain data element. One row of this table represents one data element, the data type for that data element, and a description.

Table Contents:

Data Element Name	Data Type	Key	Notes
Facebook/Email /***Google*** Username	String	100	Users have the option to log in with Facebook**,** email***,or Google***.
Facebook/Email /***Google*** Password	String	101	Users can opt to sign in with Facebook/email/***Google*** or create a password.
Drexel Username	String	102	Drexel users can sign in with Facebook or create password.
Drexel Password	String	103	Drexel users can choose to sign in with their Drexel account
Calendar	Array	104	The events and classes come from Term Master Schedule and CORQ.
Chat Room		105	Users can make chat room.
Account List		106	Users will make accounts and be able to friend other people.
Amount of Time Spent on BuddyU	Double	107	***Firebase*** stores the amount of time the user spends on BuddyU for the first couple of weeks.
Number of Downloads	Double	108	***Firebase*** stores the number of downloads of the app.
Users	String	109	***Firebase*** will store the user information.

Figure 6 - BuddyU Database Table

Figure 7-4 – Template for Detailed Design for a Code Function
Name: Functions for creating a functional code for BuddyU
Type: Function
Purpose: This function is needed to understand and begin to implement what different communicable functions we will need to make the code work.
Parameters: The following parameters are used to call this function:

Name	Data Type	Notes
RegisterEmail	***Function***	***This function will have the code to allow a user to register themselves with the BuddyU app if they do not already have an account.***
SigninGoogle	***Function***	***This function will check to see if the user entered the right information for their google account using the Firebase authentication API. If the information is correct, it will sign-in the user.***
SigninFacebook	***Function***	***This function will check to see if the user enter the right information for their Facebook account using the Firebase authentication API. If the information is right, it will sign-in the user.***
SigninEmail	***Function***	***This function will check to see if the user entered the right email information after they already registered with BuddyU using the Firebase authentication API.***
CoChat	Function	This function will create the chat feature where students can communicate with each other. This function will make it so that the chat feature is connected to the main code.
ScheduleConnector	Algorithm function	This function will use if else statements and other code to create an algorithm that will look at two schedules of students and find common time for class or lunch.
Main	Function	This function will call other functions of CoChat and ScheduleConnector and other functional functions to communicate with each other and the database using MongoDB.

Return Type: ***The RegisterEmail function will not return anything (void) but will print a toast or in other words a message on the screen whether the registration was successful or not based on whether the firebaseAuth task was successful. The SigninGoogle function, SigninFacebook function, and SigninEmail will not return anything (void) but will print a toast or in other words a message on whether or not the sign-in was successful. If the task was successful, it will change to the home screen.***
The CoChat function will be a string and will return strings that the user types in and update the string variable every time another string is inputted by the user. The Schedule Connector function will be a int function that takes in different user string parameters for their schedules. It will return the times of when the user can take common classes with their friends. The main function will not return anything but will make sure that the code runs smoothly.
Processing: ScheduleConnector PsuedoCode
Take in the parameters of the string class name of the student and the other parameter of the int time that that class is taken.
Using if/else if or else statements, create a algorithm which will output times that two people can take classes at

the same time.

Return the class name and the corresponding time to the program.

Figure 7-5 – Team Capability Assessment

Capability\Teammate	Brandin	Chris	Sanobar	Sita	Briana
Android Studio	*	*	*	**	*
Firebase	*	*	*	**	*
C++	***	***	*	*	*
HTML	***	***	**	***	***
Java	*	***	**	**	*
Python	**	**	*	*	*
JavaScript	***	***	*	**	*

** The table values represent an assessment of team member capabilities. The values are:

1 – No knowledge or not much relative to the needs of this project
2 – Enough knowledge to accomplish part but not all of this project
3 – Knowledge probably sufficient for this project