

## 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., "CustomerProfile"
- 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: DragonFriends

Team: 47

Date: 2/21/2018

Type	Name	Description or Notes
Screen	Login	Login Screen for students to input their Drexel Credentials. This is where Authentication happens.
Screen	Register	Allows the user to sign up, and initiate their profile in the User Database. Interacts Directly with the User Manager/User Database.
Screen	Profile	Display of Student information. EX, classes they are enrolled in, Age, Email, Year at College (Freshman, Sophomore, ...). Directly Interacts with the User Manager we will make.
Screen	Search	Allows students to add classes that they have, interacts with the TMS manager.
Screen	Settings	allows the user to control privacy settings on their profile, change the password and other things. Interacts with the User manager, and this data is stored under the User Database
Screen	Other Profile	Similar to the Profile screen, but with limited functionality based off of the privacy settings.
Screen	Class Roster	Allows students to see the other users in classes they are enrolled in. Interacts with the Class Database
Database	User Database	Stores the information for users in the program. Works with the User Manager.
Database	Class Database	Stores the information for classes in the program. Works with the TMS Manager, and the TMS API.
Manager	Authentication Manager	Works with Authent]]ication by FireBase, and the App itself. Allows Users to register/login safely with a Drexel ID.
Manager	TMS Manager	Works with the TMS API, the Class Database and the App itself. The main functionality of the Search requirement, allows students to find and register under classes that they want.
Manager	User Manager	Works with the User Database, and the App itself. The main functionality is to call/store the information of student profiles, and allow interaction between the two elements.

## Figure 7-2 – Template for Detailed Design for a Screen

**Name:** Login Screen

**Type:** Screen

**Purpose:** This screen is needed to meet requirement 3.1: Authentication

**Description:** Figure 1 shows the layout for this screen. This screen is the login screen for DragonFriends. This is where the authentication happens.

The screen contains the following elements The Screen Selection in the top left corner, used to switch between all of the different screens. The top right button allows you to exit the app without difficulty. The top two buttons will stay consistent throughout the program. There are two lines of user input, the Username and Password. This is where the user will input their designated username and password (respectively). When the Username and Password are filled in, the person will be able to press the login or sign up button. the Login button will work the FireBase authenticator to allow them to login. The Sign Up button will allow them to create their user profile. The last button is the Forgot Password Button, allowing the user to re-authenticate their profile with FireBase.

Layout:



**Figure 1 - Login Screen**

**Name:** User Profile


**Type:** Screen

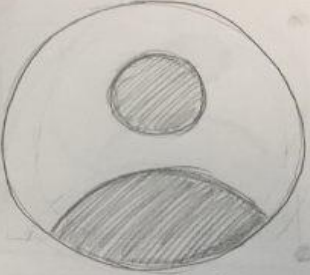
**Purpose:** This screen is needed to meet requirement 2.1: User Interface.

**Description:** Figure 2 shows the layout for this screen. This screen is used to display the details of user.

The screen contains the following elements A series of displays to show the user's information. Including a display for Email, Name, Phone Number, and Enrolled Classes. We wanted each Class to be a button, able to connect to that classes' corresponding class roster screen.

Layout:

View Profile 



✉ EMAIL: john.doe@drexel.edu

👤 NAME: John Doe

📞 PHONE NUMBER: +44 012 - 345 - 6789

🎓 ENROLLED CLASSES:

Current Term	Next Term
MATH 117	COOP 101
CHEM 103	
ENGL 103	
BIO 102	
PSY 101	
ITAL 102	

**Figure 2 - Profile Screen**

**Name:** User Settings

**Type:** Screen

**Purpose:** This screen is needed to meet requirement 2.1: User Interface

**Description:** Figure 3 shows the layout for this screen. This screen is used to allow the user to manually change their profile settings. also allow people to augment their privacy settings.

The screen contains the following elements All of the profile settings will have some combination of a input and selector or slider button, allowing them to change the details of their profile. This includes, a Name Editor, a Email Editor, Password Re-Authenticator, A Privacy Slider, and Sounds/Notification Slider. The exception we have is the profile image editor. This allows the user to add a profile image of their choice to their profile. This aspect we foresee as a V2.0 edit rather than a V1.0.

Layout:



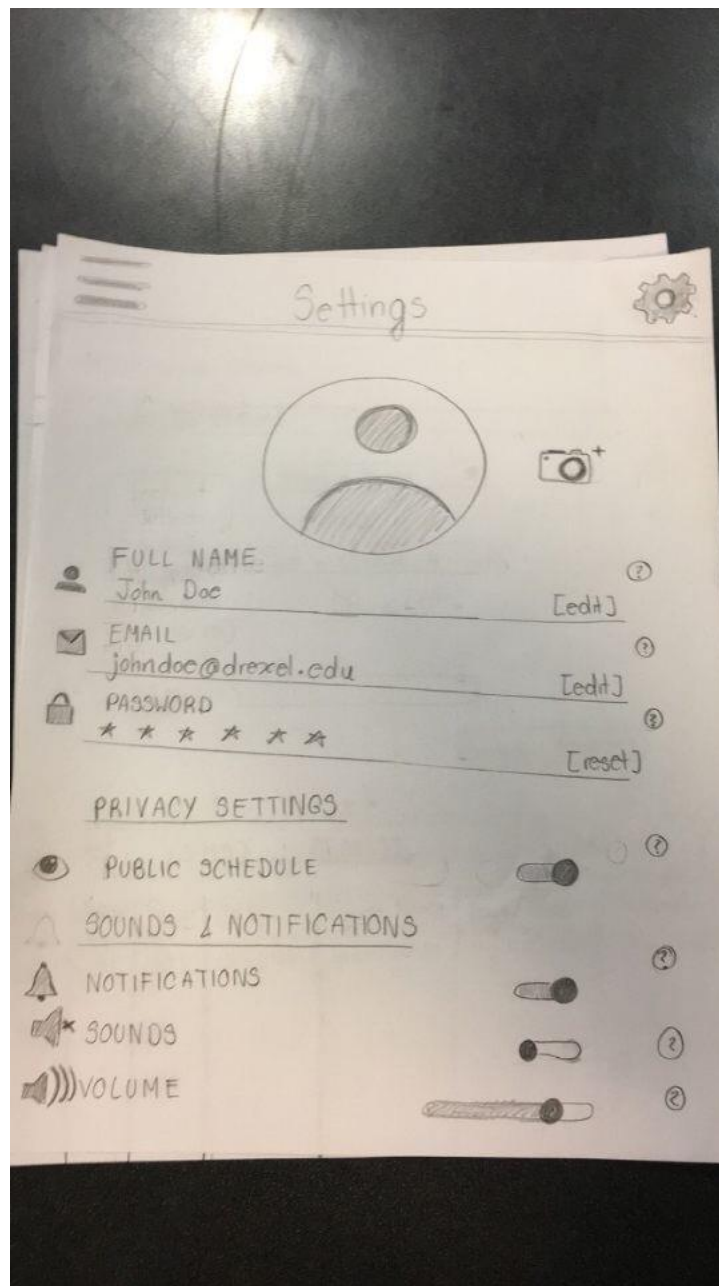


Figure 3 - Setting Screen

**Name:** Class Search

**Type:** Screen

**Purpose:** This screen is needed to meet requirement 3.2: Searching and Adding Classes

**Description:** Figure 4 shows the layout for this screen. This screen allows the user to search for and add a class.

The screen contains the following elements, a series of buttons to help refine your search, the Select Term button, the Course Name and Number entry line, and the CRN. Not all three will be required, but at least one will be needed to refine your search. The last couple of buttons are at the bottom of the screen. These buttons will be the classes themselves, allowing you to select the class, and if you press the button, “Add Course to Schedule”, that class will be added to your profile.

Layout:

**Search**

Enter Search Criteria

Search for Classes

TERM  (i)

AND

COURSE NAME & NUMBER (I.E. PSY 101)

OR

CRN

...

☐ select (TMS) **RESULTS**

Subj. Code	Course No	Sec	CRN	Course Name	Days/Times	Instructor
ABCD	101	001	01234567	University 101	F 1:00-3:00	Jane Doe

**Figure 4 - Search Screen**

**Type:** Screen

**Purpose:** This screen is needed to meet requirement 3.3: View Other Students in the Same Class

**Description:** Figure 5 shows the layout for this screen. This screen displays all of the people in a Class that you are signed up for.

The screen contains the following elements, a button to select the class that you want to view, and an email button to contact the person in that class. The email, functionality is going to be a V2.0 requirement, but primarily, this screen is just a display.

Layout:

Class Roster

Course

...

NAME	EMAIL
Carter Knight	cbk52@drexel.edu
Tri Le	tnl34@drexel.edu
Alia Yeschanora	ay387@drexel.edu
Sahiti Pisupati	sp3429@drexel.edu
Victoria Young	vly25@drexel.edu

↑

[lists names/has link to their profiles when selected]

↑

can be link to their email possibly/ send email through outlook

Figure 5 - Class Roster Screen

**Figure 7-3 – Template for Detailed Design for a Database Table****Name:** User Database**Type:** Database Table**Purpose:** This table is needed to meet requirement 3.4 - 1, User Information**Description:** Figure 1 shows the contents for this table. This table shows how we store the information of each user. One row of this table represents a key-value pair of each piece of information that we store from the user.**Table Contents:**

Data Element Name	Data Type	Key	Notes
Email	String	email	
Year level	String	yearLevel	“freshman”, “sophomore”, “prejunior”, “junior”, “senior”
User Classes	Javascript Object	classes	Object of classes
- Class	Javascript Object	class	Contains information about a specific class
Phone number	String	phoneNumber	
User id	String	id	Each user has a unique randomly generated ID

**Figure 1 - User Database Table**

**Name:** Class Database

**Type:** Database Table

**Purpose:** This table is needed to meet requirement 3.1 - 3: View Other Students in the Same Class

**Description:** Figure 2 shows the contents for this table. This table shows how we store the data explaining what people are in each class. One row of this table represents a key-value pair of each piece of information we store for each class.

**Table Contents:**

Data Element Name	Data Type	Key	Notes
Classes	Javascript object	classes	
- CRN	Number	CRN	
- Course name	String	courseName	
- Roster	Javascript Object	roster	Contains IDs of every member in the class

**Figure 2 - Class Database Table**

**Figure 7-4 – Template for Detailed Design for a Code Function**

**Name:** TMS Manager

**Type:** Function

**Purpose:** This function is needed to meet requirement 3.2 - 1: & 3.1 - 2: Searching for/ Adding Classes

**Parameters:** The following parameters are used to call this function:

Name	Data Type	Notes
CRN	number	CRN is the course number

**Return Type:** JSON object of classes information

**Processing:**

TMS.getClassInfo(CRN)

Makes a GET request to the TMS API

Returns the class information for the class with that specific CRN



**Name:** Access Manager

**Type:** Function

**Purpose:** This function is needed to meet requirement 3.1 - 1: Authentication

**Parameters:** The following parameters are used to call this function:

Name	Data Type	Notes
email	String	
password	String	

**Return Type:** a unique userID from Firebase Authentication

**Processing:**

signInWithEmailAndPassword(email, password)

Returns user ID from Firebase

CI102

**Name:** User Manager

**Type:** Function

**Purpose:** This function is needed to meet requirement 3.4-1: User Information

**Parameters:** The following parameters are used to call this function:

Name	Data Type	Notes
User ID	String	

**Return Type:** user information from Firebase

**Processing:**

getUserInfo(userID)

Reference from firebase with a unique url

Returns all user info, including phone number, email, classe

**Figure 7-5 – Team Capability Assessment**

	<b>Tri Le</b>	<b>Carter Knight</b>	<b>Victoria Young</b>	<b>Alia Yeszhanova</b>	<b>Sahithi Pisupati</b>
<b>Android Studio</b>	1	1	2	2	1
<b>Java</b>	2	1	3	3	2
<b>Firebase</b>	3	1	1	1	1
<b>Server side code (TMS)</b>	2	1	1	1	1
<b>Javascript</b>	2	2	2	2	2
<b>Proto.io (for beta screens)</b>	1	1	1	2	1

\*\* 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