Class Torch:
iOS Programming
iPhone App Project Documentation

by Jae Heon Kim

Index

1. Author

This app was created by Jae Heon Kim (**jhk774**).

2. Project Title

The project title is called ***Class Torch***. I came up with the name by combining the terms '***Class***' (from NYU Classes) and '***Torch***' (inspired by our New York University logo) together. I believed that since my app contains a heavy amount of NYU Classes functionalities, this was an interesting and unique approach to coming up with a title.

3. Overview of the iPhone App

Over the past few years here at NYU, I have experienced numerous difficulties getting a full grasp of the classes that we are enrolled in. Sometimes we can feel like we are behind schedule (i.e. missing deadlines), struggling with a concept that we hardly understand, or simply left out. There can also be some cases where we miss lectures due to illnesses and / or emergencies. Sometimes there will be a group project assigned, but we are left alone without having anyone to form a group with. The professors could be too occupied to provide us immediate assistance. For the following reasons, my idea of "**Class Torch**" came together, which is a platform where classmates as well as the professors, teaching assistants, graders, tutors can collaboratively work and interact together online outside of the direct classroom environment. Students are very familiar with the term *Piazza*, a web service that provides a forum for students and instructors to communicate and collaborate. I initially pictured a *Piazza*-like system that is exclusively designed for NYU that also includes a global chat room feature that can be efficiently used by all users. To sum up, it is an all-in-one app for NYU students and instructors.

Part of the reason that I created this app is also because a lot of times students have to move around frequently in between our respective NYU resources. We have a countless number of platforms that we are required to use for academic proactivity: NYU Classes, NYU mail, Duo Mobile, Albert, Piazza, Gradescope, other external websites, etc. Although I was not able to implement highly sophisticated algorithms like the following, I believed my app contains highly proficient functionalities that can be useful in the academic environment of our university.

To give a basic rundown of my app, ***Class Torch*** provides essential NYU Classes functionalities, which are the **syllabus, gradebook and assignments** page. My app efficiently allows students to obtain information about the classes that they are enrolled in. Instructors will use my app to send out **notifications** (this includes assignments and assignment specifications, grades released, or public announcements) that students need to be aware of. Rather than having instructors write and send out a full email to the entire class, my systematic app provides an easier method to alert students with what they need to know.

Finally, there is a global chat room dedicated to each class. Messages will be chronologically displayed and anyone that is taking part of the class can freely start an open chat conversation. This suffices the need for students to receive immediate assistance by fellow students and instructors, rather than having to use for example the forums section of NYU classes or direct email. Overall, my app's motive to enhance students' time and academic management is fulfilled with our software's implementations.

4. User Instructions

When launching the app, the user will first see the ***Login Page*** which displays an option to choose between student user and instructor user. However, if you do not have an account already created within my database server, the user must press the ***Sign-Up*** button to create an account. The ***Sign-Up Page*** will allow users to select their role here at NYU (either student or instructor), choose their school (CAS, Tandon, Stern, Tisch, etc.) and create a username or password. By doing so, all this information will be saved within my database **tables** which I will talk about more under the **Compilation and Installation Instructions (5)** part of this report. Pressing create account will bring the user back to the Login page. After logging in, student users and instructor users will see their respective **Universal Landing Page**. I differentiated the two ViewControllers by adding different gradient colors for student user and instructor users. The Landing Page will display three buttons: **Notifications Board, NYU Classes, Chat Room**. Students will see all the notifications sent out by the instructor under their notifications board, while for instructors instead of the button being notifications board, it will be **Send Notifications**. The **NYU Classes** button will lead to the **NYU Classes Page**, which will display three buttons: **Syllabus, Assignments, and Gradebook**. Again, these buttons are the essential NYU Classes functionalities that I have talked about earlier in my iPhone App overview. Students can view the class syllabus, assignments required for the class, and their grades under the gradebook. Instructors will upload the class syllabus, send out assignments and assignment specifications, and finally release grades. Finally, the Chat Room will chronologically display all the messages sent out to the entire class. Instructors are also able to monitor what is being said on the chat room which will be used mostly amongst students.

As a side note, when creating an account, if the password the user has created (for both student and instructor) is already preexistent within my database server (meaning a different user is already using that password), the created account will not be saved in my database tables. Therefore, while running the app, the user must make sure to check the console on XCode. The console will display **User Not Found or Incorrect Password!!** if the user tries to login with a non-existent account.

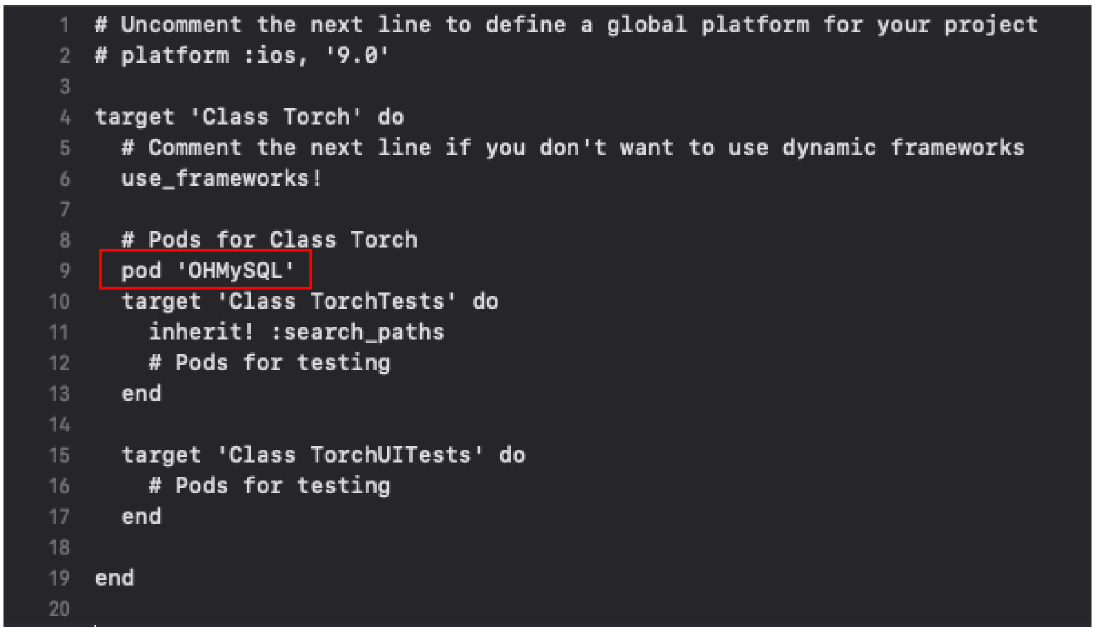
5. Compilation and Installation Instructions

Before I talk about the compilation and installation instructions, I must briefly talk about **MySQL** and **OHMySQL API**. MySQL is based off a client-server model where one client requests resources from my server. MySQL acts as a database system which contains all my data. For my app, information stored in my server includes student name, password, and school / department. Instead of storing all this information about the users under one large form of memory, the database stores data in its own distinct **tables**. Some of the tables I have created include **students, professors, notifications** and so forth. MySQL is an open source, meaning anyone can use and change the software.

Because I am using MacOS for this project, we must know the fact that in Apple SDK, there is no API that can directly connect to the MySQL server. Therefore, for my project, I am using **OHMySQL API** which allows us to connect to my MySQL database. Here are the steps:

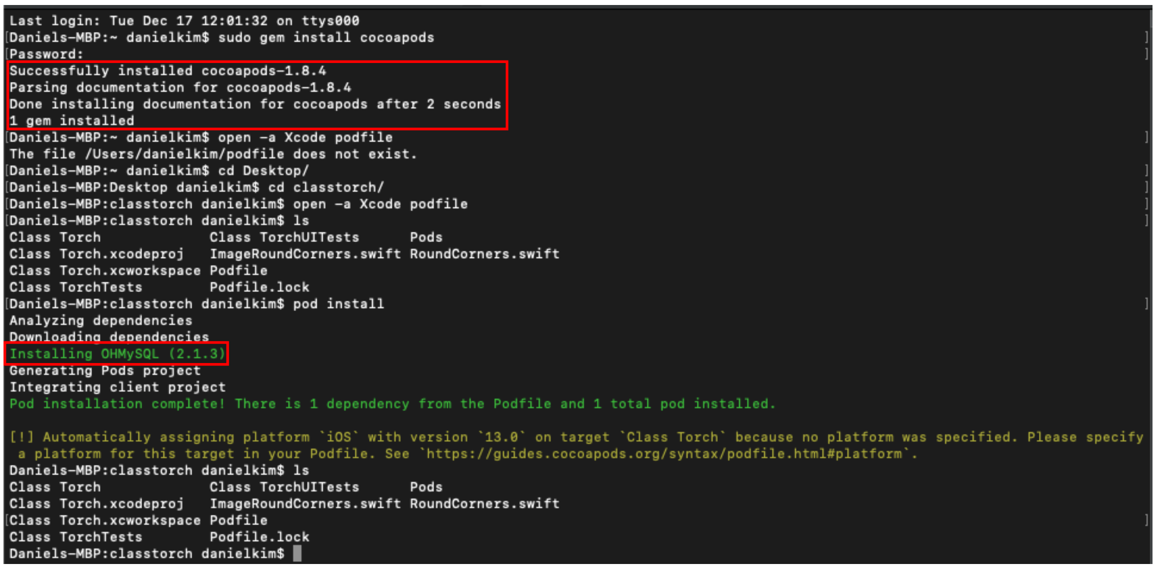
I first install CocoaPods by opening our terminal. The command is **sudo gem install cocoapods**. After typing our password for our computer, the terminal will display **Successfully**

installed cocoapods-1.8.4 and **Parsing documentation for cocoapods-1.8.4**. After doing so, I will go to my directory where my project file ***classtorch*** is located. We type in cd Desktop/ and then cd classtorch/. We initialize / create a podfile typing **pod init**. Once that is done, I go back to the classtorch folder, and now I can open the podfile from the terminal. We type in **open -a XCode podfile**. The reason why I am doing this is because I must add a command to install the **OHMySQL API**. It is very simple: the command is **pod 'OHMySQL'**. I will type this under # Pods for Class Torch. Now for the final installation, I type **pod install**. The terminal will analyze and download dependencies, and then successfully install the OHMySQL. Below are the screenshots of the steps I took to install the OHMySQL API using **CocoaPods**.



```
# Uncomment the next line to define a global platform for your project
# platform :ios, '9.0'

target 'Class Torch' do
  # Comment the next line if you don't want to use dynamic frameworks
  use_frameworks!

  # Pods for Class Torch
  pod 'OHMySQL'
  target 'Class TorchTests' do
    inherit! :search_paths
    # Pods for testing
  end

  target 'Class TorchUITests' do
    # Pods for testing
  end

end

```

- Rectangle: I typed in the command pod 'OHMYSQL' under # Pods for Class Torch.



```
Last login: Tue Dec 17 12:01:32 on ttys000
Daniels-MBP:~ danielkim$ sudo gem install cocoapods
Password:
Successfully installed cocoapods-1.8.4
Parsing documentation for cocoapods-1.8.4
Done installing documentation for cocoapods after 2 seconds
1 gem installed
Daniels-MBP:~ danielkim$ open -a Xcode podfile
The file /Users/danielkim/podfile does not exist.
Daniels-MBP:~ danielkim$ cd Desktop/
Daniels-MBP:Desktop danielkim$ cd classtorch/
Daniels-MBP:classtorch danielkim$ open -a Xcode podfile
Daniels-MBP:classtorch danielkim$ ls
Class Torch             Class TorchUITests      Pods
Class Torch.xcodeproj   ImageRoundCorners.swift RoundCorners.swift
Class Torch.xcworkspace Podfile
Class TorchTests        Podfile.lock
Daniels-MBP:classtorch danielkim$ pod install
Analyzing dependencies
Downloading dependencies
Installing OHMySQL (2.1.3)
Generating Pods project
Integrating client project
Pod installation complete! There is 1 dependency from the Podfile and 1 total pod installed.

[!] Automatically assigning platform `iOS` with version `13.0` on target `Class Torch` because no platform was specified. Please specify
 a platform for this target in your Podfile. See `https://guides.cocoapods.org/syntax/podfile.html#platform`.
Daniels-MBP:classtorch danielkim$ ls
Class Torch             Class TorchUITests      Pods
Class Torch.xcodeproj   ImageRoundCorners.swift RoundCorners.swift
Class Torch.xcworkspace Podfile
Class TorchTests        Podfile.lock
Daniels-MBP:classtorch danielkim$
```

- 1st Rectangle: I have successfully installed Cocoapods to our system.
- 2nd Rectangle: I have successfully installed OHMySQL API using Cocoapods.

Finally, another important aspect to my project is that instead of running the app under "**Class Torch.xcodeproj**", I have to run it by clicking **"Class Torch.xcworkspace"**. In order to launch the application appropriately with full functionalities installed through cocoapods, you need to execute **"Class Torch.xcworkspace"** instead of "**~.xcodeproj**".

6. APIs Called

As mentioned above, the API called is the **OHMySQL API**. For my MySQL server, I used a AWS EC2 instance for my server. The EC2 instance creates virtual server in Amazon's Elastic Compute Cloud. It provides a variety of instance types with different configurations of CPU, memory, storage and networking resources. (The MySQL Server Version is 5.7, our ip is 13.125.254.61, our port is 3306 and our database name is NYU_class) Again, I used this API because there is no default SDK for the iOS to connect to the MySQL server directly. It was installed through Cocoapods. (pod 'OHMySQL')

7. Sources of existing code incorporated in *Class Torch*

We used the Open Source on GitHub to install the OHMySQL API. Here is the attached link: https://github.com/oleghnidets/OHMySQL. I used some of the core aspects of the existing code for my project to connect to the database by setting up configurations and execute queries. Here is the code:

#import <OHMySQL/OHMySQL.h> → we first import OHMySQL API.

```
        OHMySQLUser *user = [[OHMySQLUser alloc]
                initWithUserName:@"nyu"
                        password:@"NYUdb!234"                //Database Password
                                serverName:@"13.125.254.61"  //Our IP
                                        dbName:@"NYU_class" //Database Name
                                                port:3306   //Port
                                          socket:@"/Applications/MAMP/tmp/mysql/mysql.sock"];
```

We set up our configurations

```
OHMySQLStoreCoordinator *coordinator = [[OHMySQLStoreCoordinator alloc]
initWithUser:user];
[coordinator connect];
```

We connect using coordinator

```
OHMySQLQueryContext *queryContext = [OHMySQLQueryContext new];
queryContext.storeCoordinator = coordinator;
OHMySQLQueryRequest *query = [OHMySQLQueryRequestFactory SELECT:@"students"
condition:nil];
```

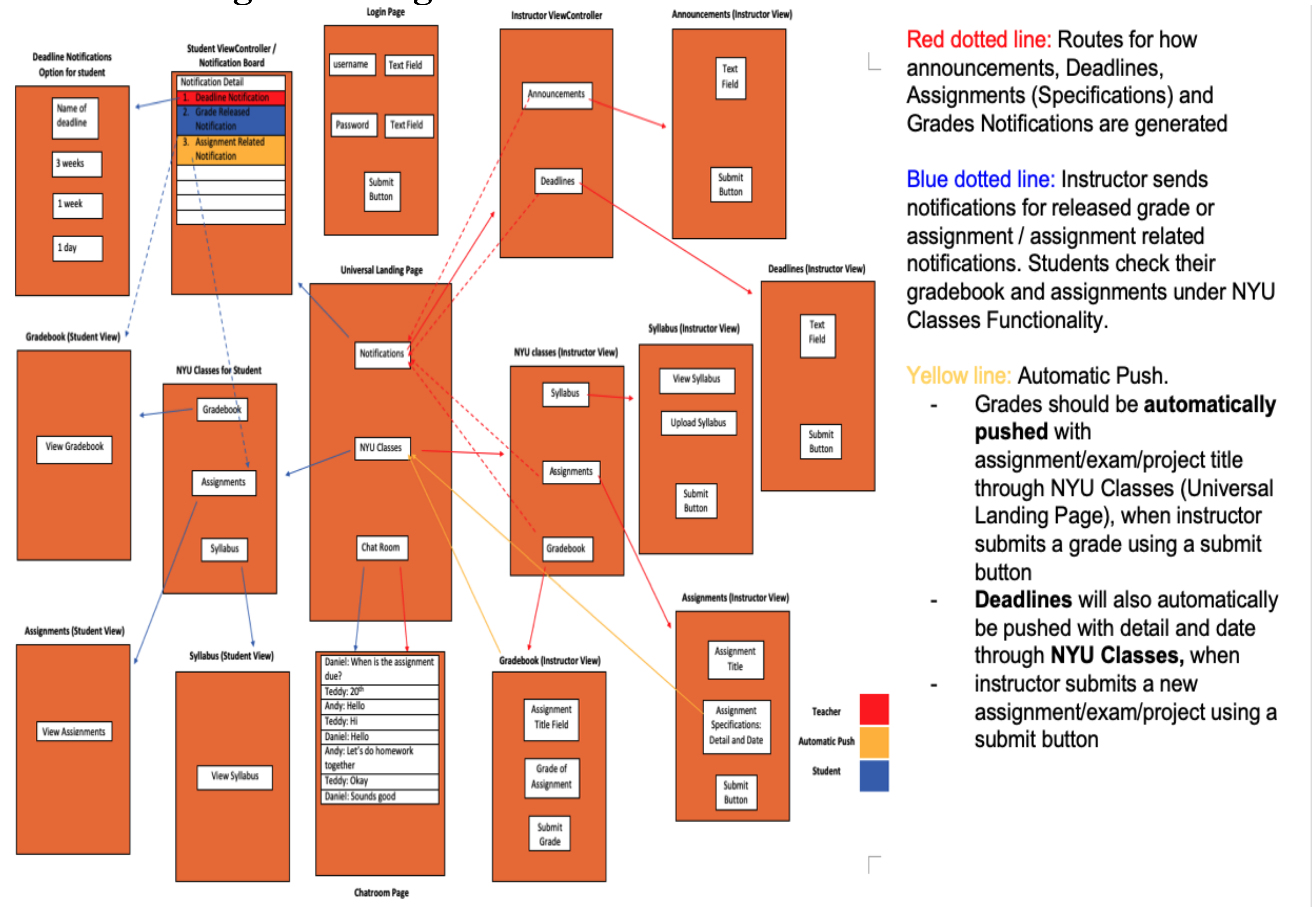
We set query context to execute our queries, and then select from whichever table we would like. In this case, our query (request) is from our **students** table.

```
OHMySQLQueryRequest *query = [OHMySQLQueryRequestFactory INSERT:@"chat"
set:@{ @"content": _input.text }];
NSError error = nil;
```

Here we are inserting data into our table, in this case **chat** table that contains all our chatroom messages, and setting the content (text).

This is the most fundamental aspect of my app. Simply put, all the tables that require storage of data (**students, professors, chat room, notifications**) is structured by following along these lines of code.

8. Overall Logic and High Level Flowchart



Daniel: When is the assignment due?
Teddy: 20th
Andy: Hello
Teddy: Hi
Daniel: Hello
Andy: Let's do homework together
Teddy: Okay
Daniel: Sounds good

Red dotted line: Routes for how announcements, Deadlines, Assignments (Specifications) and Grades Notifications are generated

Blue dotted line: Instructor sends notifications for released grade or assignment / assignment related notifications. Students check their gradebook and assignments under NYU Classes Functionality.

Yellow line: Automatic Push.

- Grades should be **automatically pushed** with assignment/exam/project title through NYU Classes (Universal Landing Page), when instructor submits a grade using a submit button
- **Deadlines** will also automatically be pushed with detail and date through **NYU Classes,** when
- instructor submits a new assignment/exam/project using a submit button

We will start with the login page which is on top of the *Universal Landing Page*. Student user or Instructor user will login with their account information. It will then lead to the *Universal Landing Page* which contains three buttons: *Notifications, NYU Classes and Chat Room*. The Notifications Button will lead to the Notifications Board for the student, while instructors will be able to send out notifications. I will further describe the flowchart starting with the student ViewControllers on the left side.

Students will be able to see all their notifications (*deadlines, announcements, assignment specifications*) under the Notifications Board. I am yet to implement the *deadline notification ViewController* for students to use. The idea was that whenever a student presses one of the deadline notifications under the notifications board, it will lead to the *Deadline Notification option for student* ViewController. Here, students will be able to choose between 3 weeks, 1 week, 1 day to get deadline notifications.

NYU Classes simply contains Gradebook, Assignments and Syllabus. Students will check their syllabus, released grades and assignments by pressing one of the buttons under this ViewController. Finally, the *Chat Room ViewController* will display chronologically all the chats that have been sent out.

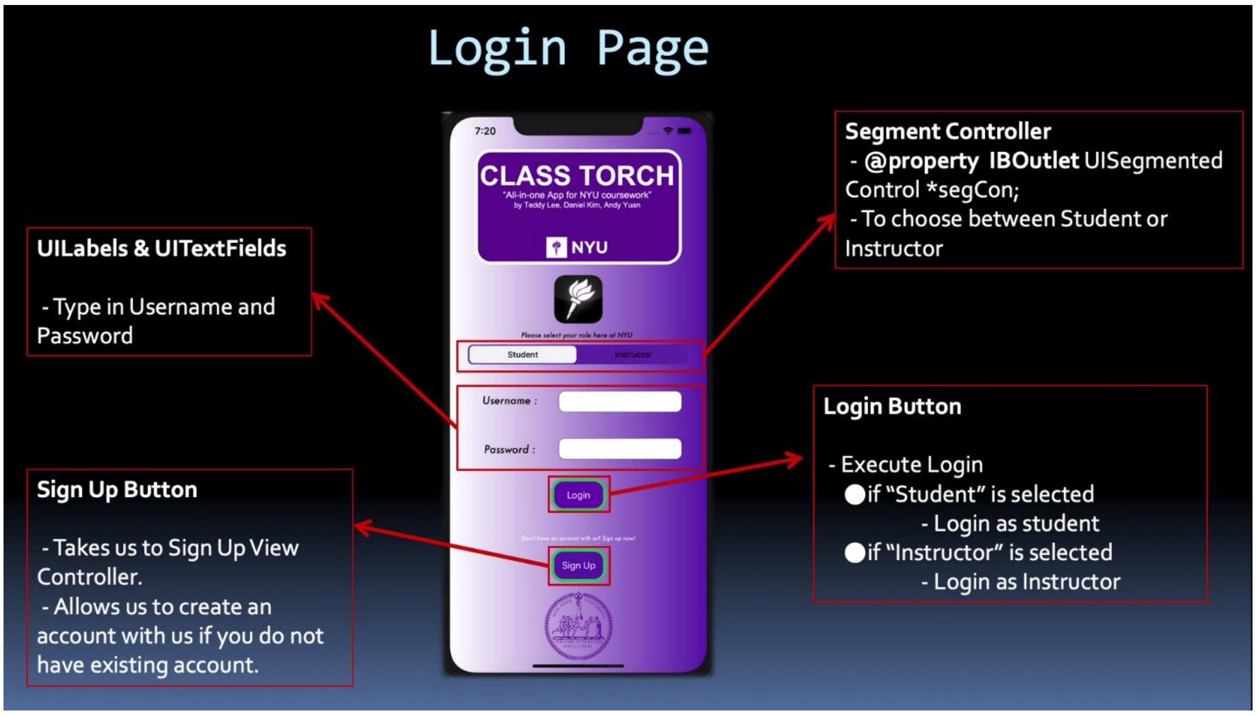
For the instructor, pressing the notifications button will lead to the *Notifications ViewController for Instructors* page. Instructors can choose between Announcements and

Deadlines as the type of notification to send out. These options will lead to its respective ViewControllers (*Assignments & Deadlines*) where the Instructor can simply type the title of announcement / deadline and the context. Instructors will finalize and send them out pressing the submit button.

The *NYU Classes ViewController* for the Instructor will also display Syllabus, Assignments and Gradebook. The Instructor can upload the syllabus, send out assignment specifications and release grades. Instructors can also view and join the Chat Room.

9. Screenshots and Logo

The logo is inspired by the NYU Torch logo which symbolizes our university. The tip of the torch (other end of the flame) is shaped as a pen and the overall logo is tilted sideways, representing academic proficiency and studiousness. Below are the screenshots of some of the most important pages of my app.



Login Page

UILabels & UITextFields

- Type in Username and Password

Sign Up Button

- Takes us to Sign Up View Controller.
- Allows us to create an account with us if you do not have existing account.

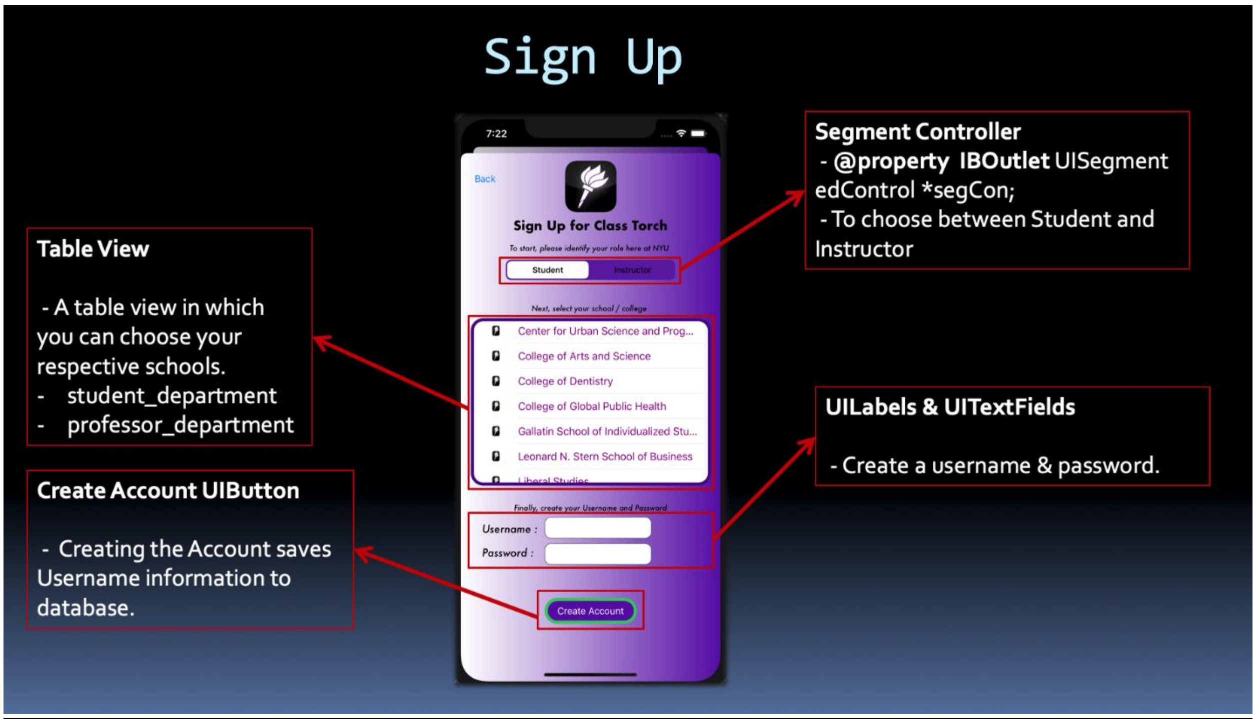
Segment Controller

- @property IBOutlet UISegmented Control *segCon;
- To choose between Student or Instructor

Login Button

- Execute Login
  - if "Student" is selected
    - Login as student
  - if "Instructor" is selected
    - Login as Instructor

7:20
CLASS TORCH
"All-in-one App for NYU coursework"
by Teddy Lee, Daniel Kim, Andy Yuan
NYU
Please select your role here at NYU
Student
Instructor
Username :
Password :
Login
Don't have an account with us? Sign up now!
Sign Up



Sign Up

Segment Controller
- **@property IBOutlet** UISegment edControl *segCon;
- To choose between Student and Instructor

Table View

- A table view in which you can choose your respective schools.
- student_department
- professor_department

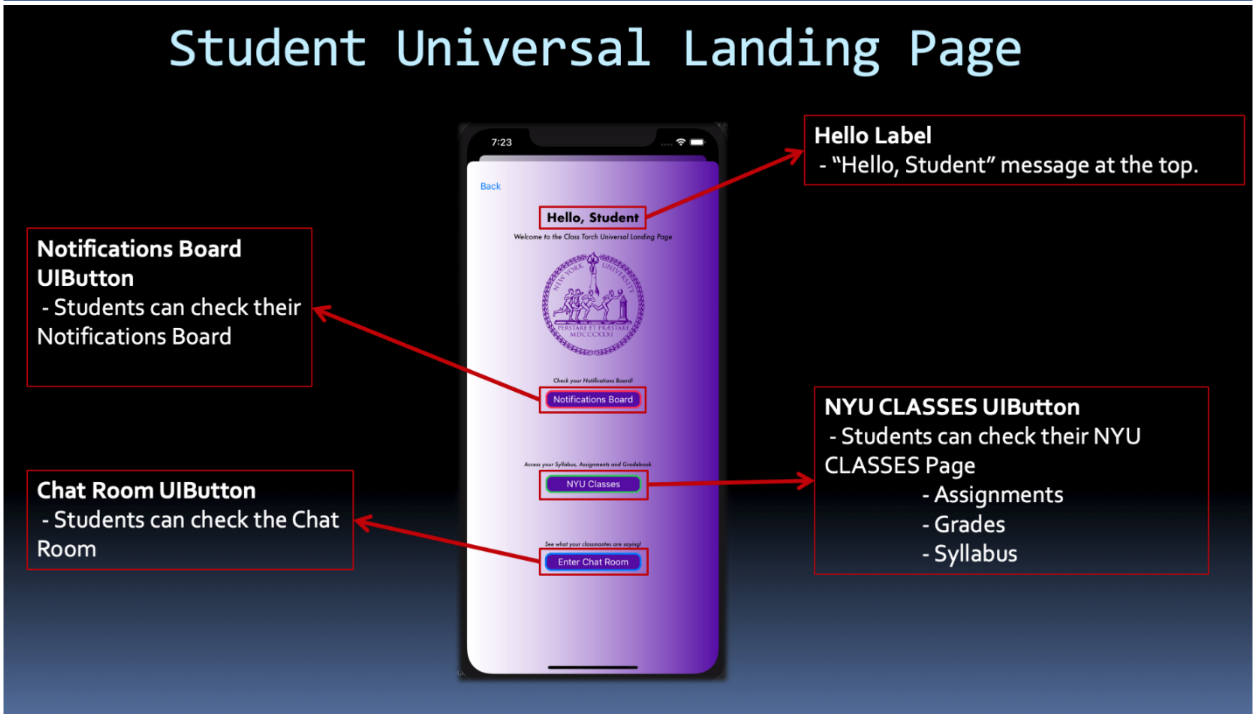
UILabels & UITextFields

- Create a username & password.

Create Account UIButton

- Creating the Account saves Username information to database.

7:22
Back
Sign Up for Class Torch
To start, please identify your role here at NYU
Student
Instructor
Next, select your school / college
Center for Urban Science and Prog...
College of Arts and Science
College of Dentistry
College of Global Public Health
Gallatin School of Individualized Stu...
Leonard N. Stern School of Business
Liberal Studies
Finally, create your Username and Password
Username :
Password :
Create Account



Student Universal Landing Page

Hello Label
- "Hello, Student" message at the top.

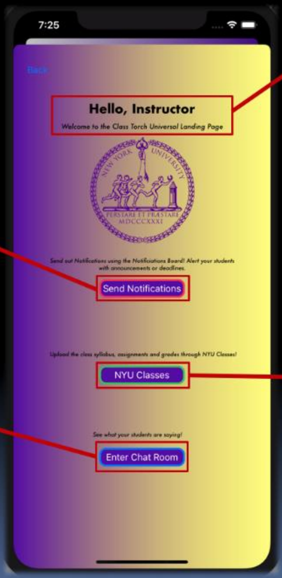
Notifications Board UIButton
- Students can check their Notifications Board

NYU CLASSES UIButton
- Students can check their NYU CLASSES Page
  - Assignments
  - Grades
  - Syllabus

Chat Room UIButton
- Students can check the Chat Room

7:23
Back
Hello, Student
Welcome to the Class Torch Universal Landing Page
Check your Notifications Board!
Notifications Board
Access your Syllabus, Assignments and Gradebook
NYU Classes
See what your classmates are saying!
Enter Chat Room

Instructor Universal Landing Page


7:25

Back

Hello, Instructor

Welcome to the Class Torch Universal Landing Page

Send out Notifications using the Notificiations Board! Alert your students with announcements or deadlines.

Send Notifications

Upload the class syllabus, assignments and grades through NYU Classes!

NYU Classes

See what your students are saying!

Enter Chat Room

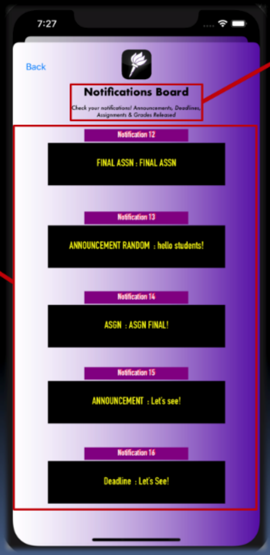
Hello Label
- "Hello, Instructor" message at the top.

Notifications UIButton
- Unlike students who just see the notifications board, instructor can send notifications

Chat Room UIButton
- Instructor can enter Class Chat Room

NYU CLASSES UIButton
- Professors can upload / adjust NYU CLASSES material:
    - Upload Assignments & Assignment specifications
    - Upload Syllabus
    - Upload Gradebook

Notifications Board


7:27

Back

Notifications Board

Check your notifications! Announcements, Deadlines, Assignments & Grades Released

Notification 12

FINAL ASSN : FINAL ASSN

Notification 13

ANNOUNCEMENT RANDOM : hello students!

Notification 14

ASGN : ASGN FINAL!

Notification 15

ANNOUNCEMENT : Let's see!

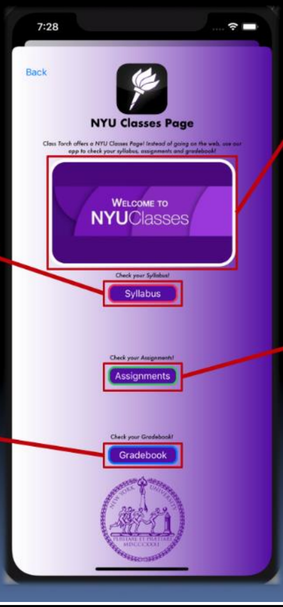
Notification 16

Deadline : Let's See!

Notifications Board Label
- Title of Notification Board

List of Notifications
- Notifications are displayed in chronological order, latest notification at the last.

NYU Classes



7:28

Back

NYU Classes Page

Class Torch offers a NYU Classes Page! Instead of going on the web, use our app to check your syllabus, assignments and gradebook!

Check your Syllabus!

Check your Assignments!

Check your Gradebook!

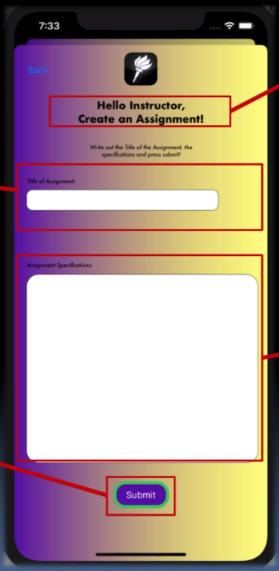
UIImage
- Welcome to NYU Classes!

Syllabus UIButton
- Students can check the course syllabus

Assignments UIButton
- Students can check their assignments sent out by Instructor

Gradebook UIButton
- Students can check their gradebook sent out by Instructor

Assignments



7:33

Back

Hello Instructor,
Create an Assignment!

Write out the Title of the Assignment, the specifications and press submit!

Title of Assignment:

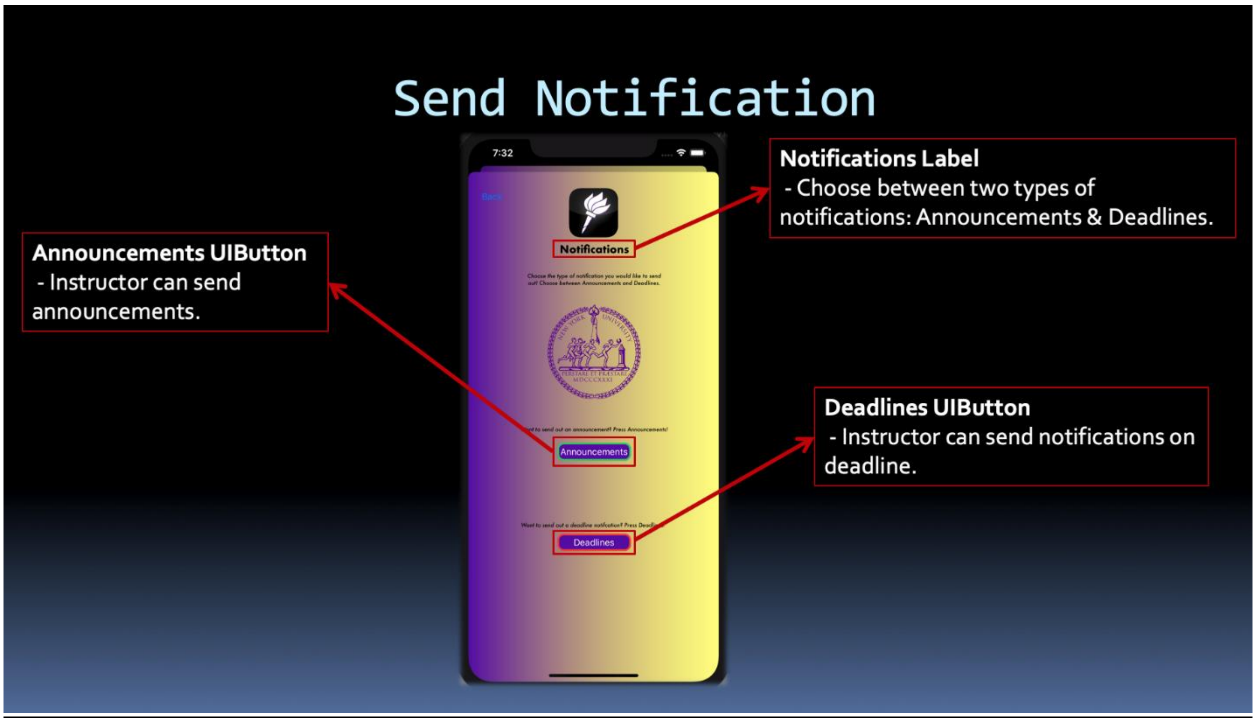
Assignment Specifications:

Hello Label
- "Hello Instructor, Create an Assignment!" message at the top.

Title of Assignment Label & Text Field
- Instructor can write Title of Assignment.

Assignment Specifications Label & Text Field
- Instructor can write specifications to assignment.

Submit UIButton
- Assignment will be sent to Notifications Board and also to Assignments under NYU Classes.

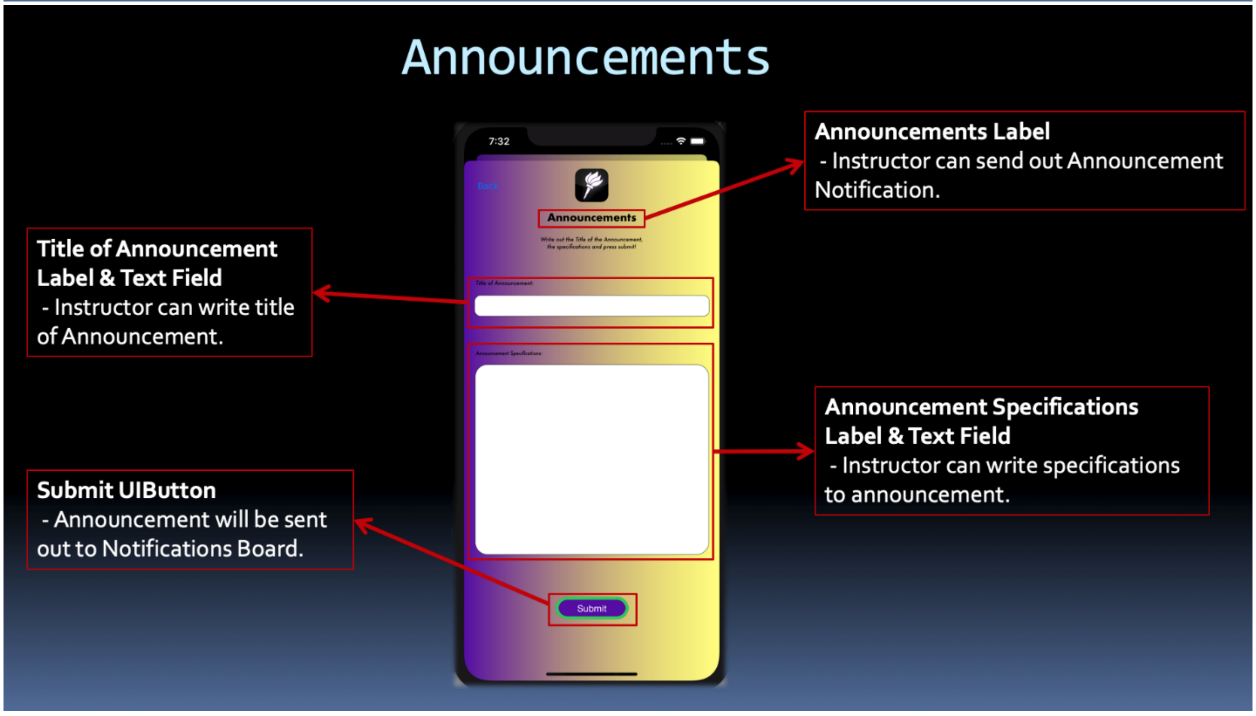


Send Notification

Notifications Label
- Choose between two types of notifications: Announcements & Deadlines.

Announcements UIButton
- Instructor can send announcements.

Deadlines UIButton
- Instructor can send notifications on deadline.



Announcements

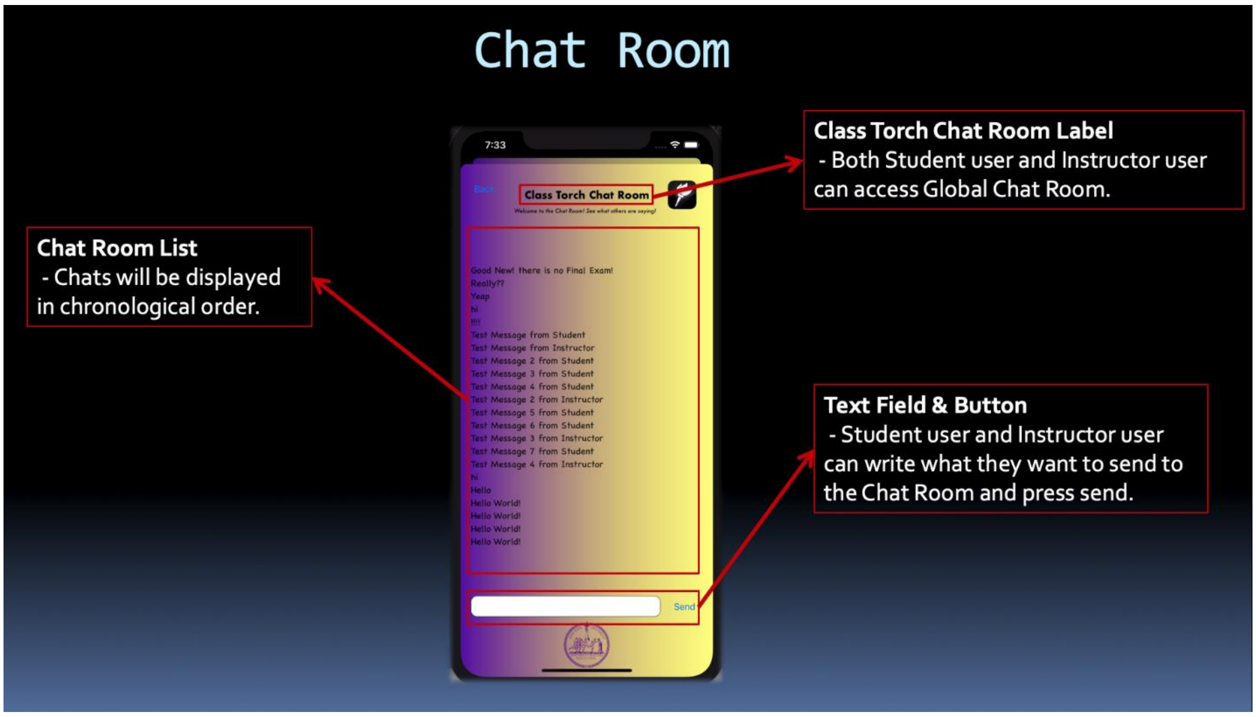
Announcements Label
- Instructor can send out Announcement Notification.

Title of Announcement Label & Text Field
- Instructor can write title of Announcement.

Announcement Specifications Label & Text Field
- Instructor can write specifications to announcement.

Submit UIButton
- Announcement will be sent out to Notifications Board.

Chat Room



Class Torch Chat Room Label
- Both Student user and Instructor user can access Global Chat Room.

Chat Room List
- Chats will be displayed in chronological order.

Text Field & Button
- Student user and Instructor user can write what they want to send to the Chat Room and press send.

Class Torch Chat Room

Welcome to the Chat Room! See what others are saying!

Good New! there is no Final Exam!
Really??
Yeap
hi
!!!!
Test Message from Student
Test Message from Instructor
Test Message 2 from Student
Test Message 3 from Student
Test Message 4 from Student
Test Message 2 from Instructor
Test Message 5 from Student
Test Message 6 from Student
Test Message 3 from Instructor
Test Message 7 from Student
Test Message 4 from Instructor
hi
Hello
Hello World!
Hello World!
Hello World!
Hello World!

10. Breakdown of Work

Created proposals, Configured AWS EC2 instance as a database (MySQL) server, Designed Database Architectures (Students, Professors, Chats, etc.), Implemented ViewControllers based on the designs by other teammates, Installed API through Cocoapods, Used API to connect Database server, Implemented business logics (login, chats, assignments, etc.) through Objective-C, Created PowerPoint presentations and presented to the class. Created app's logo, Implemented making appearances of buttons, images, text fields using Swift (gradients, buttons, etc.), took care of general design and layout of ViewControllers (as it is portrayed in the flowchart), helped create PowerPoint, presented to the class.

11. Future of the App – including posting on the Apple App Store

In the upcoming future I would like to expand the app by implementing a more specific column within our table. I can enhance the app by adding more specifications about the users themselves. Furthermore, I would like to have a broader approach by taking this platform to other universities as well. Other colleges also implement a system like NYU Albert for course registration, graduation progress and so forth. Meaning, other schools would also take advantage of a system like Class Torch to provide students more information about their work and status at their own colleges and universities.

12. Additional Features to be Implemented in the coming months

I was unable to fully implement uploading syllabus and the gradebook. I was uncertain of how exactly I could upload a Word document or PDF file using our app and send out grades. I considered these aspects to be additional features implemented in the coming months as I expand the app’s functionalities.

Nevertheless, if I were to get the university’s approval, I would like to crawl the NYU API and obtain specific data needed for my app. The most essential aspect would be allowing the app service to be provided for students enrolled in a specific class automatically rather than having to create a separate database server from our end. Students could enter a class code and automatically join the course, and their NYU Net id and password would let students to login without having to create a separate account under our database. Having the school allow us to use their API would open lots of doors for our project in the coming months. Some other features could include a ‘point system’ for users who use my app to support not only themselves but other classmates. For example, the Chat Room feature of my app allows users to gain the academic help they need while being part of a sociable environment outside of the classroom. If one user actively participates more than others, that user could get a point reward system under their account.

As for the UI designs for this project, I focused more on functionalism and ease-of-use. Aesthetic considerations were a bit overlooked, as I used a lot of basic default UI elements as opposed to designing my own. In the future, I wish to design my own set of UI element graphics such as buttons (specific designs, including both "when pressed" graphics and "when not pressed" graphics). I want to use less elements that seem to be placed on top of the background, which makes it look a bit unnatural; I want to make texts blend into the background to make it look as if they were all in a level surface/plane-- in other words: less use of colored boxes that surround texts. Furthermore, I want to look more into more professional, elegant, and modern fonts. I also wish to use drop-down menus for the notifications board to distinguish between notification types: assignments, announcements, and deadlines. There is room for improvement in the chatroom as well, such as user IDs juxtaposed with the corresponding chat messages for users to be able to tell who sent the message, which will allow for a more fluid conversation flow and for instructors to be able to easily track/monitor it. All in all, my utmost goal with the aesthetics is to make it look simple, modern, and sleek.