iLETU

Software Design Document

Version 1.1

December 11, 2013

Prepared for

COSC 4853 - Software Engineering II

LeTourneau University

2100 S Mobberly Ave.

Longview, TX 75607

Table of Contents

1. Scope

1. System objectives

2. Hardware, software, and human interfaces

3. Major software functions

4. Externally defined database

5. Major design constraints, limitations

2. Reference Documents

1. Sources for existing software documents

2. Documentation for the system you are building

3. Sources for technical reference documents

3. Design Description

1. Data description

1. Review of data flows

2. Review of data structures

2. Derived program structure

3. Interfaces within the structure

4. Entities

1. Views

2. Supporting Entities

5. File structure and global data

1. External file structure

1. Logical structure

2. Logical record description

3. Access method

2. Global data

3. File and data cross reference

6. Test Provisions

1. Test guidelines

2. Integration strategy

3. Special considerations

7. Packaging

1. Special program overlay provisions

2. Transfer considerations

8. Special Notes

9. Appendices

A. Class Diagram

B. Use Cases

C. Miscellaneous Charts and Diagrams

1. Scope
   1. System objectives

This document describes the design of the iLETU iPhone app. iLETU was originally created by Dustin Masters, but it is in severe need of an update. The goal of this project is to improve on the original design, add new features, fix bugs, and incorporate LeTourneau branding.

* 1. Hardware, software, and human interfaces

This software shall run on an iPhone or iPad. iLETU shall use a storyboard for the human interface of both versions.

* 1. Major software functions

The software shall have 11 distinct purposes. These purposes take the form of views in the application. All requirements are related to one or more specific views. All classes with the exception of the App Delegate and the View Controllers can be traced to one or more specific views. The views are as follows:

* About
* Announcements
* Campus Bullet
* Calendar
* Dining Services
* Floor Plans
* Hours
* Intramurals
* Online
* Course Schedule
* Yellowjacket News

See Section 4.1, Views, for more information.

* 1. Externally defined databases

The application shall access the following types of external data:

* RSS Feeds
* HTML Pages

Some of the pages shall require authentication through a HTTP challenge, Javascript, or HTML Post. See Section 5, Views, for information on each specific View and Appendix B – Use Case Diagram for an overview of each protocol.

* 1. Major design constraints, limitations

The application shall follow current Apple development human interface standards:

* + 1. Deference – The UI helps users to understand the content and never competes with it.
       1. Take advantage of the whole screen

Don’t feel the need to put content in boxes; consider letting it stretch across the whole page

* + - 1. Reconsider visuals that add realism

Stay away from unnecessary effects such as shadows that clutter the display

* + 1. Clarity – Focus on functionality and keep it simple. Use visuals carefully and make sure icons and text are crisp and lucid.
       1. Use negative space effectively

Negative space makes the app seem more efficient and draws attention to important content

* + - 1. Use the system fonts for readability
      2. Embrace borderless buttons

Use appropriate titles, context, and color to indicate interactivity

* + 1. Depth – Use layers and motion to heighten the user’s experience
       1. Use transitions to emphasize relationships
       2. Make important content pop out into the foreground

1. Reference Documents
   1. Sources for existing software documents
   2. Documentation of system
2. iLETU SRS Version 1. 2013-09-02.
3. Jesse Andersen. JASidePanels. Github. 2013-11-07. <<https://github.com/gotosleep/JASidePanels>>.
4. NimbusKit. NimbusKit. 2013-09-02. <<http://nimbuskit.info/>>.
   1. Sources for technical reference documents
5. iOS Developer Library. Designing for iOS 7. 2013-09-18. <<https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/index.html#//apple_ref/doc/uid/TP40006556>>.
6. Drew Sikora. Using PDL for Code Design and Documentation. Gamedev.net. 2001-06-13. < <http://www.gamedev.net/page/resources/_/technical/general-programming/using-pdl-for-code-design-and-documentation-r1384>>.
7. Design Description
   1. Data description
      1. Review of data flows

The iLETU app is driven by the letuiphoneAppDelegate. This class shall have methods that are called when the app is opened, terminated, put in the background, and brought to the foreground. When the app is opened, the application:didFinishLaunchingWithOptions: method shall be called and two panels shall be created. The first panel is on the left side of the screen but completely hidden behind the second panel. This panel serves as a navigation menu that creates and displays the front panels as described in Sections 1.3 and 4.1.

At any time, the user may press a button on the top-left corner of the screen. This shall reveal the hidden panel, and the user may choose a new menu item to navigate to.

* + 1. Review of data structures

The panel for each menu item shall be created via storyboard. Each panel shall also have a corresponding class that implements the required panel methods and the proper delegator for the class. For more information, see Section 4.1, Views.

The app shall also use the following non-view data structures:

1. Announcement.m – Stores a single entry from an RSS feed
2. FilteredURLCache.m – Caches a webpage, while stripping away any ads
3. constants.h – Stores constants that are used throughout the app.
   1. Derived program structure

The program shall create the hidden left panel and initialize the right panel as the About view. When the user triggers a view change, the program shall create the corresponding view if it doesn’t already exist and display it in the right panel.

* 1. Interfaces within the structure

All main views are required to inherit from the UIViewController class.

Constants shall be stored in a header file, constants.h. This includes any strings that will be displayed and any URLs. The required URLs are as follows:

* Bus schedule - http://www.letu.edu/opencms/opencms/homepage-links/student-resources/abbottbus.lnk
* Pool hours - http://www.letu.edu/\_Other-Resources/solheim-hours.html
* YAC calendar - http://www.google.com/calendar/embed?showTitle=0&showCalendars=0&showTabs=0;&showPrint=0&height=600&wkst=1&bgcolor=%23EEEBDB&src=yac%40letu.edu&color=%231B887A&ctz=America%2FChicago
* Athletics page - http://www.letuathletics.com/index.aspx
* Announcements - http://letustartpage.blogspot.com/feeds/posts/default?alt=rss
* Campus Bullet - http://campusbullet.net/home/category/all/?feed
* Yellowjacket - http://letuyellowjacket.org/feed/
* Saga menu - http://legacy.cafebonappetit.com/rss/menu/147
* Intramurals schedule - http://www.imleagues.com/School/Portal.aspx?SchID=39d7b8cb08db465ab4f27a061607a7c4&Portal=SchOfDay
* Mail - https://mail.letu.edu/owa/auth/logon.aspx
* Web services - https://my.letu.edu/ICS/
* Blackboard - https://courses.letu.edu
* Student directory - http://www.letu.edu/search/student\_directory.html
* Chapel attendance - http://www.letu.edu/student\_life/chapel/attendance.html

1. Entities
   1. Views
      1. Announcements
         1. Processing narrative with cross reference to SRS

This view inherits functionality from the RSSView class, Section 4.2.1.

When the Announcements View is activated by the Application Delegate, the software shall connect to the announcements RSS feed, which will be stored in constants.h. The software shall load each article into the Announcement data structure, which has a field for the title, date, and body of the article. See Section 4.2.1.1 for more detail about the feed and list.

* + - 1. Interface description

The Announcements view inherits interfaces from the RSSView class detailed Section 4.2.1.2. The RSSView class inherits from the Nimbus Library.

* + - 1. Design language (or other) description

When the view is opened:

* + - * 1. Set the title of the window to “Announcements”
        2. Set the URL of the RSS feed to the one listed in the constants file
        3. Follow the normal steps to set up an RSS viewer with the URL and download the articles
        4. For each article, put the title and date in a list
      1. Modules used

This view inherits from the RSSView.

* + - 1. Data organization

This class shall depend on a delegate to control its flow. The delegate shall handle transitions between views. The style of the delegate shall change depending on if the device is in iPhone or iPad. It shall also inherit data organization from the RSSView class, Section 4.2.1.5.

* + - 1. Comments

None

* + 1. Online
       1. Processing narrative with cross reference to SRS

When the Announcements View is activated by the Application Delegate, the software shall present the user with the following options:

1. Configure Auto-Login
2. Email
3. Blackboard
4. Web Services
5. Student Directory
6. Chapel Attendance

When the user selects the first option, Configure Auto-Login, the app shall request the user’s LETNET credentials. It will then verify the credentials by sending a login request to the Student Directory webpage, which will be stored in constants.h. If the connection is successful, the app shall save the credentials. The other options will each open a webpage using the Nimbus library’s HTML viewer.

For Web Services and Blackboard, the app will use an HTML Post to submit the data. For the Student Directory and Chapel Attendance, the app will use an HTTP challenge. For email, the app shall use Javascript to login. The user will be able to return to the main menu at any time by clicking a button on the Navigation bar.

* + - 1. Interface description

The Online View inherits from the ViewController class.

* + - 1. Design language (or other) description

When the view is opened:

* + - * 1. Set the title of the window to “Online”
        2. Display each button
        3. When an option is pressed, get the appropriate URL from the constants file
        4. Load the page from the URL
        5. The user may navigate to any link, go to the previous page, go to the next page, or go back to the original page.
      1. Modules used

This view uses the ShortcutWebViewController.

* + - 1. Data organization

This class shall depend on a delegate to control its flow. The delegate shall handle transitions between views.

* + - 1. Comments

None

* + 1. Calendars
       1. Processing narrative with cross reference to SRS

This view inherits functionality from the RSSView class, Section 4.2.1.

When the Calendars View is activated by the Application Delegate, the software shall connect to the announcements RSS feed, which will be stored in constants.h. The software shall load each article into the Announcement data structure, which has a field for the title, date, and body of the article. See Section 4.2.1.1 for more detail about the feed and list. Additionally, the Calendars view shall allow the user to view the YAC calendar and athletics page.

* + - 1. Interface description

The Calendars view inherits interfaces from the RSSView class detailed Section 4.2.1.2.

* + - 1. Design language (or other) description

When the view is opened:

* + - * 1. Set the title of the window to “Calendar”
        2. Set the URL of the RSS feed to the one listed in the constants file
        3. Follow the normal steps to set up an RSS viewer with the URL and download the articles
        4. For each article, put the title and date in a list
        5. If the user presses the YAC or Athletics button, hide the RSS feed and display the page at the URL listed in constants.h
        6. If the user then presses the Calendar button, display the RSS feed again
      1. Modules used

This view inherits from the RSSView.

* + - 1. Data organization

This class shall depend on a delegate to control its flow. The delegate shall handle transitions between views. The style of the delegate shall change depending on if the device is in iPhone or iPad. It shall also inherit data organization from the RSSView class, Section 4.2.1.5.

* + - 1. Comments

None

* + 1. Dining Services
       1. Processing narrative with cross reference to SRS

This view inherits functionality from the RSSView class, Section 4.2.1.

When the Dining Services View is activated by the Application Delegate, the software shall connect to the announcements RSS feed, which will be stored in constants.h. The software shall load each article into the Dining Services data structure, which has a field for the title, date, and body of the article. See Section 4.2.1.1 for more detail about the feed and list.

* + - 1. Interface description

The Dining Services view inherits interfaces from the RSSView class detailed Section 4.2.1.2.

* + - 1. Design language (or other) description

When the view is opened:

* + - * 1. Set the title of the window to “Dining Services”
        2. Set the URL of the RSS feed to the one listed in the constants file
        3. Follow the normal steps to set up an RSS viewer with the URL and download the articles
        4. For each article, put the title and date in a list
      1. Modules used

This view inherits from the RSSView.

* + - 1. Data organization

This class shall depend on a delegate to control its flow. The delegate shall handle transitions between views. The style of the delegate shall change depending on if the device is in iPhone or iPad. It shall also inherit data organization from the RSSView class, Section 4.2.1.5.

* + - 1. Comments

None

* + 1. Hours
       1. Processing narrative with cross reference to SRS

When the Hours view is activated, it displays the hours that buildings on campus are open for. All of the hours are hard coded, except the bus schedule and the pool hours; they shall each have a button that opens the appropriate URL from constants.h. The user shall then press the back button to return to the previous view.

* + - 1. Interface description

The Hours View inherits from the ViewController class.

* + - 1. Design language (or other) description

When the view is opened:

* + - * 1. Set the title of the window to “Campus Hours”
        2. Display the hours of each building on a button.

If the user presses the “Bus Schedule” or “Pool Hours” button:

* + - * 1. Set the URL to the one listed in the constants file
        2. Show the page corresponding to the URL
        3. Wait for the user to press the back button and return to the previous menu
      1. Modules used

There are no Modules used

* + - 1. Data organization

The required data shall be stored in the constants.h file.

* + - 1. Comments

None

* + 1. Campus Bullet
       1. Processing narrative with cross reference to SRS

This view inherits functionality from the RSSView class, Section 4.2.1.

When the Campus Bullet View is activated by the Application Delegate, the software shall connect to the announcements RSS feed, which will be stored in constants.h. The software shall load each article into the Announcement data structure, which has a field for the title, date, and body of the article. See Section 4.2.1.1 for more detail about the feed and list.

* + - 1. Interface description

The Campus Bullet view inherits interfaces from the RSSView class detailed Section 4.2.1.2.

* + - 1. Design language (or other) description

When the view is opened:

* + - * 1. Set the title of the window to “Campus Bullet”
        2. Set the URL of the RSS feed to the one listed in the constants file
        3. Follow the normal steps to set up an RSS viewer with the URL and download the articles
        4. For each article, put the title and date in a list
      1. Modules used

This view inherits from the RSSView.

* + - 1. Data organization

This class shall depend on a delegate to control its flow. The delegate shall handle transitions between views. The style of the delegate shall change depending on if the device is in iPhone or iPad. It shall also inherit data organization from the RSSView class, Section 4.2.1.5.

* + - 1. Comments

None

* + 1. Yellowjacket
       1. Processing narrative with cross reference to SRS

This view inherits functionality from the RSSView class, Section 4.2.1.

When the Yellowjacket View is activated by the Application Delegate, the software shall connect to the announcements RSS feed, which will be stored in constants.h. The software shall load each article into the Announcement data structure, which has a field for the title, date, and body of the article. See Section 4.2.1.1 for more detail about the feed and list.

* + - 1. Interface description

The Yellowjacket view inherits interfaces from the RSSView class detailed Section 4.2.1.2.

* + - 1. Design language (or other) description

When the view is opened:

* + - * 1. Set the title of the window to “Yellowjacket”
        2. Set the URL of the RSS feed to the one listed in the constants file
        3. Follow the normal steps to set up an RSS viewer with the URL and download the articles
        4. For each article, put the title and date in a list
      1. Modules used

This view inherits from the RSSView.

* + - 1. Data organization

This class shall depend on a delegate to control its flow. The delegate shall handle transitions between views. The style of the delegate shall change depending on if the device is in iPhone or iPad. It shall also inherit data organization from the RSSView class, Section 4.2.1.5.

* + - 1. Comments

None

* + 1. About
       1. Processing narrative with cross reference to SRS
          1. The About View shall display information about the app and LeTourneau University and give a link to the LeTourneau Youtube channel, which can be found in constants.h
       2. Interface description

The About view does not inherit from any interfaces

* + - 1. Design language (or other) description

When the view is loaded:

* + - * 1. Display a short description of the app.
        2. If the user presses the “About LeTourneau University” button, instead display information about the university and a line to the LeTourneau University Youtube channel.
      1. Modules used

No modules are used.

* + - 1. Data organization

The information about the app and university shall be stored in constants.h

* + - 1. Comments

None

* + 1. Intramurals
       1. Processing narrative with cross reference to SRS

When the Intramurals View is activated by the Application Delegate, the software shall connect to the designated URL, which will be stored in constants.h. The software shall display the page in the application.

* + - 1. Interface description

The Intramurals view inherits from the ViewController class.

* + - 1. Design language (or other) description

When the view is opened:

* + - * 1. Set the title of the window to “Intramurals”
        2. Set the URL to the one listed in the constants file
        3. Show the page corresponding to the URL
      1. Modules used

There are no modules used

* + - 1. Data organization

This class shall depend on a delegate to control its flow. The delegate shall handle transitions between views. The style of the delegate shall change depending on if the device is in iPhone or iPad.

* + - 1. Comments

None

* + 1. Schedule
       1. Processing narrative with cross reference to SRS

When the Schedule View is activated by the Application Delegate for the first time, it shall connect to the designated URL, which will be stored in constants.h Once the user enters all of their classes, the software shall save the URL and open that URL from then on.

* + - 1. Interface description

The Schedule view inherits from the ViewController class

* + - 1. Design language (or other) description

When the view is opened:

* + - * 1. Set the title of the window to “Schedule”
        2. Set the URL to the one listed in the constants file
        3. Show the page corresponding to the URL

When the user finishes entering their classes:

* + - * 1. Save the new URL and replace the one listed in the constants file
      1. Modules used

There are no modules used

* + - 1. Data organization

This class shall depend on a delegate to control its flow. The delegate shall handle transitions between views. The style of the delegate shall change depending on if the device is in iPhone or iPad.

* + - 1. Comments

None

* + 1. Floor Plan
       1. Processing narrative with cross reference to SRS

The Floor Plan view shall display the floor plan of each classroom building. The app shall display a list of buttons with the building names on them that show the corresponding floor plan when pressed:

* Longview Hall
* Glaske
* Education
* HHH
  + - 1. Interface description

The Floor Plan view inherits from the ViewController class.

* + - 1. Design language (or other) description

When the view is opened:

* + - * 1. Display the Longview Hall image
        2. The user may zoom using native multi-touch functions
        3. If the user presses another button, display the floor plan associated with that building
      1. Modules used

No modules are used by this view

* + - 1. Data organization

The images shall be stored on the device.

* + - 1. Comments

None

* 1. Supporting entities
     1. RSSView class
        1. Processing narrative with cross reference to SRS

This entity provides the framework to download and display RSS feeds. It takes a URL and has methods that download the feed. After the feed is finished downloading, each item is put in a single column table view. The articles shall all be initially marked as unread by putting a dot next to the item. As soon as an unread item is opened, the item should be marked as read and the dot removed. The list shall then disappear, and the selected article shall display on the screen until the user presses the back button which shall send the user back to the list. If the user presses the refresh button on the upper right corner of the screen, the app shall re-download all of the information and mark all articles as unread.

* + - 1. Interface description

The RSSView shall interface with the controller through a set of delegates, and each delegate requires the view to implement predefined functions. The delegates are listed below, and they shall be addressed in more detail in a later version of this document.

* UITableViewDelegate
* NSXMLParserDelegate

It shall also have the following methods to interface with its parent container:

* (void)refreshContent
* (void)appendWithURL:(NSString\*)url
* (void)screenLoading:(BOOL)loading
  + - 1. Design language (or other) description

When loading data from an RSS feed:

* + - * 1. Attempt to connect to the feed.
        2. If the connection fails, show an error explaining the problem.
        3. Else, for each article download its title, link, date, and summary.
        4. Put the articles in a list ordered chronologically, newest at the top
        5. Mark all articles as unread

When the refresh button is pressed:

1. Empty the list of articles.
2. Load the data from the feed again.

When an article is selected from the list:

1. Transition to a page showing the article title, link, date, and summary.

When the back button is pressed while viewing an article:

1. Transition back to the list of articles
2. Mark the article viewed as read
   * + 1. Modules used

This class is a subclass of UIViewController, a standard class in the iOS library.

* + - 1. Data organization

The RSS reader parses one article at a time and adds it to an array. This class shall not be used directly in the app but has many subclasses that are used in views. These are the views that inherit from RSSView:

* Announcements
* Calendar
* Campus Bullet
* Yellowjacket
  + - 1. Comments

This class is never used directly, but many of the views are subclasses of this one. The RSSView leaves the decision about the content displayed in the table to its subclasses.

1. File Structure and Global Data
   1. External File Structure
      1. Logical Structure

The data that shall be pulled from an external source shall be stored as a standard RSS feed. Standard tags shall be used in order to facilitate the parsing of the file later. Tags shall be nested in order to group related information.

* + 1. Logical Record Description

The RSS feeds shall all contain several sections, including the title of the element, the date that it was posted on, the link that it forwards to, and the body containing the actual information for each element.

* + 1. Access Method

The RSS feed shall be accessed and downloaded by the device. Once it is stored locally, it shall be parsed for information, and the elements shall have their title displayed in a list that is sorted by date. Each element shall have all parsed associated information stored with it. When an element is selected from the list, a new screen shall open up displaying the body content.

* 1. Global Data

For the auto-login online services, the username and password of the user shall be saved in the user preferences storage area of the device. When they need to be accessed to sign in to the different online services, the pages shall be able to retrieve these strings and use them to authenticate the user.

* 1. File and Data Cross Reference

There shall be no data that shall be modified from multiple locations. The only stored data that shall be modified is the login information, and it shall only be modified from the auto-login page. RSS files shall only be stored on a per-session basis. They shall be overwritten when the refresh button is pressed.

1. Test Provisions
   1. Test guidelines

The software shall be tested both online and offline. The features that require an internet connection are required not to crash the app during offline use. Features shall each be tested by demonstration on different devices and iOS versions. They are required to work as intended.

* 1. Integration strategy

The views shall each be tested on the simulator, independently from any other features. Once they are verified, they shall all be integrated and tested on the simulator. Finally, the software shall be tested on all available devices.

* 1. Special considerations

N/A

1. Packaging
   1. Special program overlay provisions

N/A

* 1. Transfer considerations

N/A

1. Special Notes

The software shall use the JASidePanel and Nimbus modules. JASidePanel gives the functionality to have a hidden panel that controls navigation. Nimbus is an iOS framework that simplifies the iOS development process by adding powerful tools for common tasks.

1. Appendices
   1. Appendix A – Class Diagram

constants.h

AutoLoginView

AboutView

FloorView

1

1

YellowjacketView

DiningServicesView

loadingOverlayView

CampusCalendarView

CampusBulletView

AnnouncementsView

RSSView

ShortcutView

ScheduleView

IntramuralsView

HTMLView

LandscapeAboutLETUView

LandscapeAboutAppView

AboutLETUView

AboutAppView

Appirater

LeftNavigationView

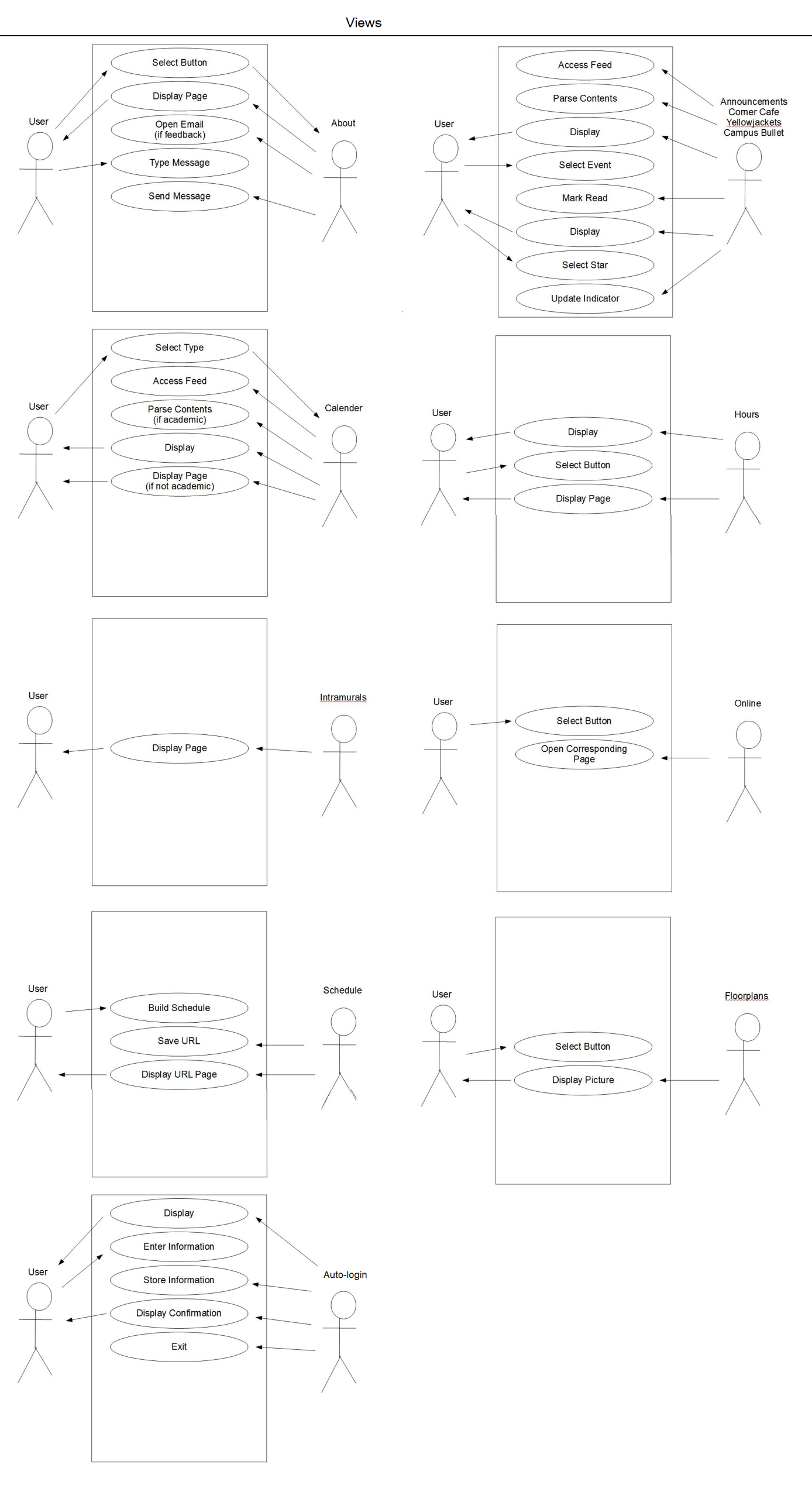
LETUViewController

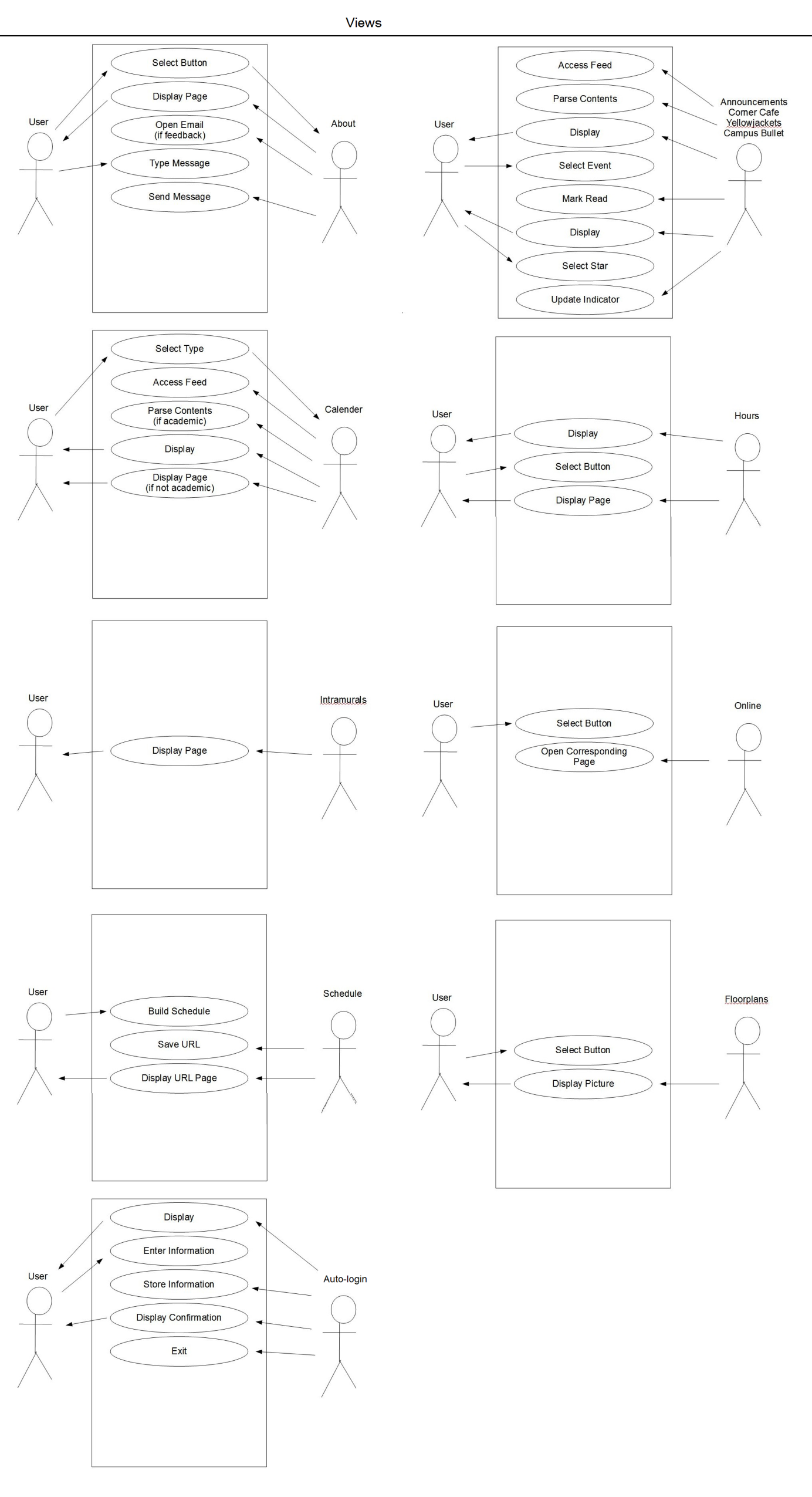
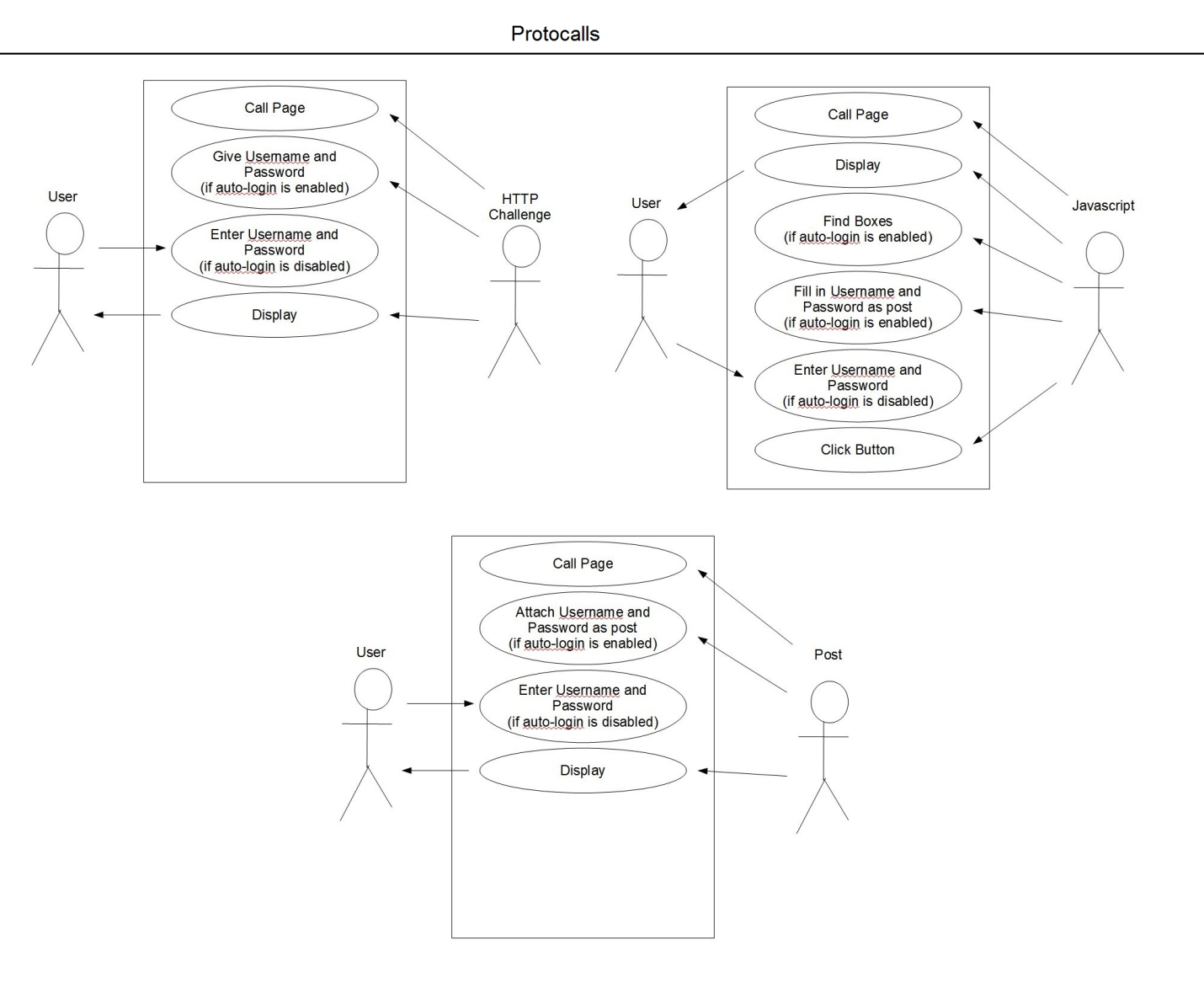
letuiphoneAppDelegate

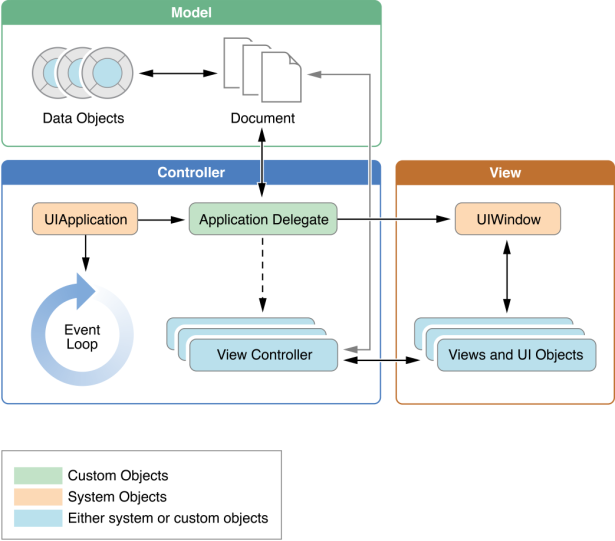
mainStoryboard\_iPad

mainStoryboard\_iPhone

main

* 1. Appendix B – Use Case Diagram



* 1. Appendix C – Miscellaneous Charts and Diagrams
     1. Model Controller View Diagram