STEM Innovation Project 2017:
NightCap: A Student-Centered Mobile Platform for Improved Campus Safety

John Bowllan (2019)

Gloria Breck (2018)

Melody Cheung (2020)

Elias Guerra (2018)

Devon Player (2018)

Joey Hernandez (2019)

Josh Ravishankar (2020)

Jay Silverstein (2019)

Dr. Noah Graham, Professor of Physics
Dr. Frank Swenton, Professor of Mathematics
Dr. Jeremy Ward, Professor of Biology

Middlebury College
June-August 2017

Contents

1. Executive Summary

Every year, STEM Innovation collaborates to find an innovative solution to a local problem. For the last five years, every year that this program has been sponsored, groups have considered problems involving party safety. This fact demands the improvement of party culture at Middlebury College. In light of the results of the most recently published report from the Task Force on Alcohol and Social Life, we gathered feedback from the student body through surveys, a focus group, and interviews with student leaders. In doing so, we've validated several concerns related to the College's alcohol policy, with a strong disapproval of the intrusion of campus security and the process by which transfers occur to sober friends. In addition, we reviewed the job description for the Party Monitors employment opportunity and brainstormed reasons that the program failed to gain student buy-in.

From our understanding, students are subject primarily to authority figures in times that they are vulnerable from drinking, when they would generally feel more comfortable receiving help from a trained student such as a ResLife staff member or an EMT. While the Department of Public Safety does employ their own initiative for peer-to-peer help, students have also expressed concerns that this system is unsafe. The Sober Friend Agreement often permits drunk, untrained students to be sober friends, and there is also no accountability system in place to ensure the student's safety once the sober friend is left to care for the inebriated student. Our solution to the lack of safe peer assistance in the nightlife culture is NightCap, an iphone application, which functions primarily by facilitating connections between inebriated students and trained students that will act as sober friends. The objective of our program is multi-faceted. Considering the apparent truth that students will drink regardless of any alcohol policy in place, STEM Innovation aims to make peer assistance more accessible at parties in a way that isn't dangerous. In doing so, party culture at Middlebury will be safer, and the student community will be strengthened.

STEM Innovation proposes a program that implements iOS software to facilitate communication amongst peers trained to respond in nonurgent situations. Through the production of iOS software featuring geolocation-detection and direct messaging, students and Public Safety will be able to communicate with trained assistants who respond to requests made through the application. This program will enhance the Sober Friend Agreement and make it more accessible to students, while giving Public Safety access to peers better trained to take care of another student. NightCap assistants will be able to respond to student calls and remain present until involved parties mutually end the session. To aid the app in further fostering a culture of safety consciousness, we offer adhesive hardware: a stand-alone, Bluetooth-enabled ethanol sensor which continually samples ambient air and informs users of their drinks' alcohol content. The ethanol-"sniffing" capabilities of the device may also be applied to testing one's own blood alcohol content and other potential avenues yet undetermined.

2. Background

The STEM Innovation team established an early goal of pursuing an objective that was evidently valuable in our immediate lives. As students of Middlebury College, every member of the team has been exposed to the nightlife culture of the campus. Our experiences motivated us to pursue party safety problems at Middlebury, specifically concerning unsafe recreational drinking and the risky atmosphere for personal violation. These problems are not new to Middlebury or similar colleges in the United States, the realization of which prompted the establishment of the College's Task Force on Alcohol and Social Life, comprised of faculty, staff, and students. This committee was asked by the Dean of the College to consider the impact of alcohol on the college community and to propose revisions to college policy, procedures, and infrastructure. In light of the concerns raised by this report, which were largely validated by student feedback, we dedicated the purpose of the project to improving one portion of the current Alcohol Policy that raised the most concerns with students, the Department of Public Safety's Sober Friend Agreement. In pursuit of that goal, STEM Innovation has created and is committed to constantly revising a program proposal incorporating innovative technology that establishes a better connection between students and peer resources.

Task Force on Alcohol and Social Life Final Report

While the scope of this report expands further than that of STEM Innovation, there were relevant findings pertaining to drinking culture at Middlebury. In addition, there is evidence within this report that suggests the dissatisfaction of the student body by the current alcohol policy. We found that this report provided a solid foundation for our innovation, especially in the following:

1. Relevant Findings
    a. Most drinking occurs on Friday and Saturday nights
    b. Most students drink in their residence halls (70%) with first-years doing so at an even higher rate (83%). The majority of alcohol is consumed in rooms prior to going out, which is inconsistent to national drinking patterns in which most drinking occurs off campus.
    c. First year students have a much higher rate of transfers to sober friends and emergency rooms than do other classes (Middlebury 2010 study).
    d. Almost all students have reported having been negatively impacted by alcohol on campus but would prefer the policies to stay the same or be more lenient.
    e. Students feel they must drink alcohol quickly in order to avoid getting caught by campus security. Students largely feel that more lenient policies would result in safer drinking.
    f. Social House members (and others who try to initiate social functions) feel as if they receive undue punishment for drinking on campus related to restrictions, citations, and property damage. These groups feel as if they try to comply with college policy but end up struggling with students who arrive at the events intoxicated.
2. Relevant Recommendations

a. Building a stronger sense of community and accountability among students, staff, and faculty.
b. Middlebury College Party Monitors (MCPMs): A paid student employment opportunity. Student party monitors would work closely with Public Safety to ensure the safety and well-being of all students at weekend social events. Distinct from party hosts, party monitors would be assigned to specific social settings, and will play a key role in both monitoring the setting and communicating as necessary with Public Safety.
c. Good Samaritan rule: The Task Force recommends that a Good Samaritan clause be adopted and widely communicated to students. In conjunction with our emphasis and early education on bystander intervention, students need to know that in situations where peers are clearly in need of medical assistance, they should never hesitate to call for help. Students need to know that the College's highest priority is student health and safety.

Middlebury's Policies Concerning Party Safety

Middlebury's Alcohol Policy is aimed to fully comply with local, state, and federal laws governing the possession, use, and furnishing of alcoholic beverages and controlled substances. In Compliance with the Drug-Free Schools and Community Act of 1989, Middlebury's Alcohol Policy speaks explicitly to the standards of conduct that clearly prohibit the unlawful possession, use, or distribution of illegal drugs and alcohol by students and that disciplinary sanctions will be imposed in violation of this policy. Any necessary disciplinary sanction or educational, medical, or psychological intervention enforced is also a direct result of local, state, and federal law.

In compliance with the law, Middlebury College enforces a Drug and Alcohol Policy that prohibits underage drinking, possessing alcohol in public spaces or in first-year residence halls, possession of kegs or common containers (e.g. punch bowls) without advance party registration, drinking games or any behavior designed for the purpose of becoming intoxicated, and repeated unsafe intoxication, among other dangerous acts. In the event that any of these prohibited acts are committed, a disciplinary sanction will be immediately imposed by the school.

The Sober Friend Agreement

The Sober Friend Agreement is a document supporting an initiative of the Department of Public Safety in accordance with the College's alcohol policy. This part of the policy, which speaks to situations in which students become too inebriated from alcohol consumption, calls for Public Safety to issue this agreement to another student nearby. At any point during a party gathering, anyone from another officer, to a ResLife Staff member, to another student, can raise a concern for the health of another student to a Public Safety Officer. Once a sober friend is located, they are given a GordieCheck card, which tells the symptoms of alcohol poisoning and what to do, as well as Public Safety's and Porter Hospital's phone numbers. They are then left responsible for the student for the night. This agreement enforced by Public Safety enables students to hold each other accountable for their actions in a way that emphasizes community responsibility. As per the alcohol policy of the school, if a student is assigned a Sober Friend

more than 3 times, then disciplinary action will take place in compliance with local, state, and federal law.

Good Samaritan Policy

Middlebury's Good Samaritan Policy is intended to encourage students to seek swift medical assistance for themselves and others without fear of penalty. It urges students not only to take care of their own well-being, but to behave in an equally responsible way with their peers. In times where safety concerns arise from a student's excessive drinking or drug use, students are encouraged to seek help from medical professionals or authority figures and are ensured that doing so will not result in disciplinary action. This policy refers to isolated incidents and does not excuse or protect those who flagrantly or repeatedly violate the Alcohol and Other Drugs Policy, nor does it preclude disciplinary action arising from violations of other Middlebury policies. This policy does not protect students from law enforcement personnel, but it is consistent with a law enacted by Vermont in June 2013 that provides limited immunity from prosecution to a witness or victim of a drug or alcohol overdose who seeks medical assistance to save the life of an overdose victim.

Findings from STEM Innovation

Survey 1

Student response to the program was based on two surveys posted on Facebook, both which received 100+ responses, and a dinner hosted by the STEM group for 14 students to act as a focus group. The first survey asked yes-no and short answer question about drinking and party-going habits.

- 80% of students said they drink at parties.
- 70% of students said that if a designated sober student were present they would reach out to them. The main responses for why not were:
    - they would just ask a friend,
    - they were often the sober person themselves,
    - they would not feel comfortable asking help from a stranger, or
    - they would never need help
- Students overwhelmingly said they would feel comfortable reaching out to Residential Life if they could not get help from a friend. Other responses were Public Safety and the party host.

Survey 2

The second survey asked multiple choice questions. The first question allowed selecting multiple answers.

- 50% of students said they would ask any student who had training for help, if they were drunk and couldn't find someone they knew. Other answers were a) "Any student that looked concerned" (47%), b) "A student that signed the sober friend agreement" (41%), and c) "Public Safety, even if it was not that serious" (39%).
- 18% of students said they had been appointed to be sober friends. Most did it for friends.

- 6% of students said they had received help from a sober friend and it was very helpful. 85% of students said there friends just take care of them or they don't "participate in stuff like that".
- 30% of students said they have been in situations where they would like to appoint a sober friend for someone else. 30% said they would do it themselves. 25% said they would not appoint a sober friend to someone else.
- 50% of students said they often go out sober and would be more than happy to take care of anyone that was too drunk. 17% said they would feel uncomfortable helping someone they do not know.
- 36% of students said they would be willing to be on-call to act as a sober friend for free and 36% percent said they would want to be paid. 17% said they would not do it.

Vice President of Tavern Social House: A Statement

> *My main concerns about the Sober Friend agreement revolve mainly around follow-up. At the moment, when someone is assigned a Sober Friend, they are left alone with their escort. Nobody ensures that the individual assigned as the Sober Friend brings the person requiring help back to their room, which leaves room for the possibility of assault. Yes, the person assigned to be the Sober Friend has to sign the agreement, but that does not inherently prevent them from committing acts of violence against the impaired individual. Furthermore, there is not much oversight into who can be assigned as a Sober Friend. Public Safety officers-- as busy as they are-- do not have time to screen individuals they need to act as Sober Friends. Unfortunately, then, the person acting as the Sober Friend could also be impaired, leaving the situation open to more danger.*
>
> - *Michael Schermerhorn*

Focus Group Discussion

For the dinner focus group, students gave their responses about the general design and philosophy of the program. They also tested the smartphone application. The most important student responses to the program included:

- Students would not want to participate as NightCap assistants for free. Work would take place during weekends when most students are going to parties and having fun. If the work involves taking care of drunk students it could potentially be very unpleasant and very few students would do such work for free.
- If someone wanted help for themselves they would just call or text a friend. Students are more likely to request assistance for someone else.

College Administration

Over the course of 8 weeks, the STEM team met with a number of college administrators in different departments including the Department of Health and Wellness, Public Safety, Dean of Students, Residential Life, Business Services, Parton Center for Health and Wellness, and the Risk Committee. The group also received legal advice from outside of Middlebury and it has

been included here. Not all the suggestions and critiques were consistent. Certain departments were concerned primarily with student safety while others were more concerned with liabilities. The following are some of the more important comments made:

- Consider risks to the requesters and how this could pose risks to the college. What types of problems could arise? Alcohol poisoning, sexual harassment, information privacy, etc.
- If a student used the application habitually it could indicate alcoholism or other dangerous behavior. This would likely necessitate intervention from the college in the form of counseling or disciplinary action.
- In order to maintain the safety of the students, the college would likely require some degree of oversight of the program. This would not preclude the program being student-run, but it would set boundaries for what would be allowed.
- Get the support of student group like the SGA. Get them to endorse the philosophy of the program and also get their input on the mechanics of program.
- Disclaimers, terms of service, a privacy policy, and other legal documents explicitly relieving the college and responders of legal responsibility in the event that an accident occurs will be necessary to protect from lawsuit. Responders will only be able to operate "to the best of their ability" and should not guarantee perfect service.
- The responders should travel in pairs to ensure that the responders hold each other accountable. Doing so would also keep the responders safe in case they found themselves in a dangerous situation.
- It would be helpful for the responders to make themselves available to Public Safety as on-demand sober friends. Public Safety officers are not always able to find sober students at parties to sign the Sober Friend Agreement.
- Maintaining the students anonymity was acceptable when considering the student's well being, but it could pose a big legal risk to the college.
- If responders did not report underage drinking they could be considered as enabling it. It would be necessary for respondents to report certain or all incidents and situations to the college.

3. Purpose

The duration of the 2017 STEM Innovation program consisted of fine-tuning a system through which Sober Friends could be imposed in a safer, more accessible way. Through our proposed system, students and Public Safety will be able to request assistance from trained peers that fulfill the same role as the current system. However, instead of leaving Public Safety to find a student nearby that has likely consumed alcohol themselves, officers will be able to use an iphone application to directly communicate with students much better equipped to provide support. Because this role will ideally be filled by trained, vetted students, the system incorporates a protocol for efficiently reporting each incident to the school, which enforces these assistants to be held accountable for their actions. In addition, students will have a network of peer support that they will be able to reach out to in the event that they cannot reach a reliable friend on their own in non-emergent situations, and they will also have a platform to use in cases where they wish to request intervention as a bystander. If concerns arise for a student in a party

gathering in a private space where Public Safety does not patrol, then students will now have the opportunity to notify responsible peers without having to directly contact campus security.

4. Program Development

From the beginning of the summer to the end, the main issues have been the same. As we gathered more feedback from students and Middlebury Administration certain changes to the mechanics of the program, but the overall philosophy remained the same: Create a system for students to request help from student peers when they have been drinking and feel unsafe in a way that isn't dangerous.

The original project design had an Uber-like model, where the application would serve both requesters and responders. Requesters would open the application and would have assistance sent to them. The responders would be able to sign in whenever they like and volunteer their services. For responders, minimal barriers to entry would hypothetically allow as many students as possible to assist others. We presumed that students already want to participate in assistive services, but are prevented or inhibited by logistical and social barriers. This program would have removed those barriers. As the idea developed a number of issues arose. The biggest questions over the 8-weeks in summer were:

- How do you make the program seem trustworthy to students?
- What situations would require more qualified personnel to step in?
- How qualified and trained would the responders be?
- How would the responders be chosen?
- How would the responders participation be regulated?
- What would be done with the information that was collected?
- Would the program work through the college or not?
- What legal dangers could the responders and the program face if an accident took place?
- How do students requesters and responders get paired?

Over the course of 8-weeks different answers emerged.

- To make the program attractive to students they would need to feel that they were not going to get into trouble.
- Taking care of drunk students could lead to situations involving alcohol poisoning, sexual assault, violent behavior, and other risks that could endanger the responders and the students requesters. If dangerous situations were not properly handled, then a lawsuit could occur.
- To make the program safe it would probably be necessary to let the college have some degree of oversight.
- The program should not replace or interfere with existing college safety services like Public Safety and Residential Life. It should aim to enhance the programs that already exist.
- The responders should not be handling emergencies, but instead informing more qualified parties if they occur.

5. Software Development

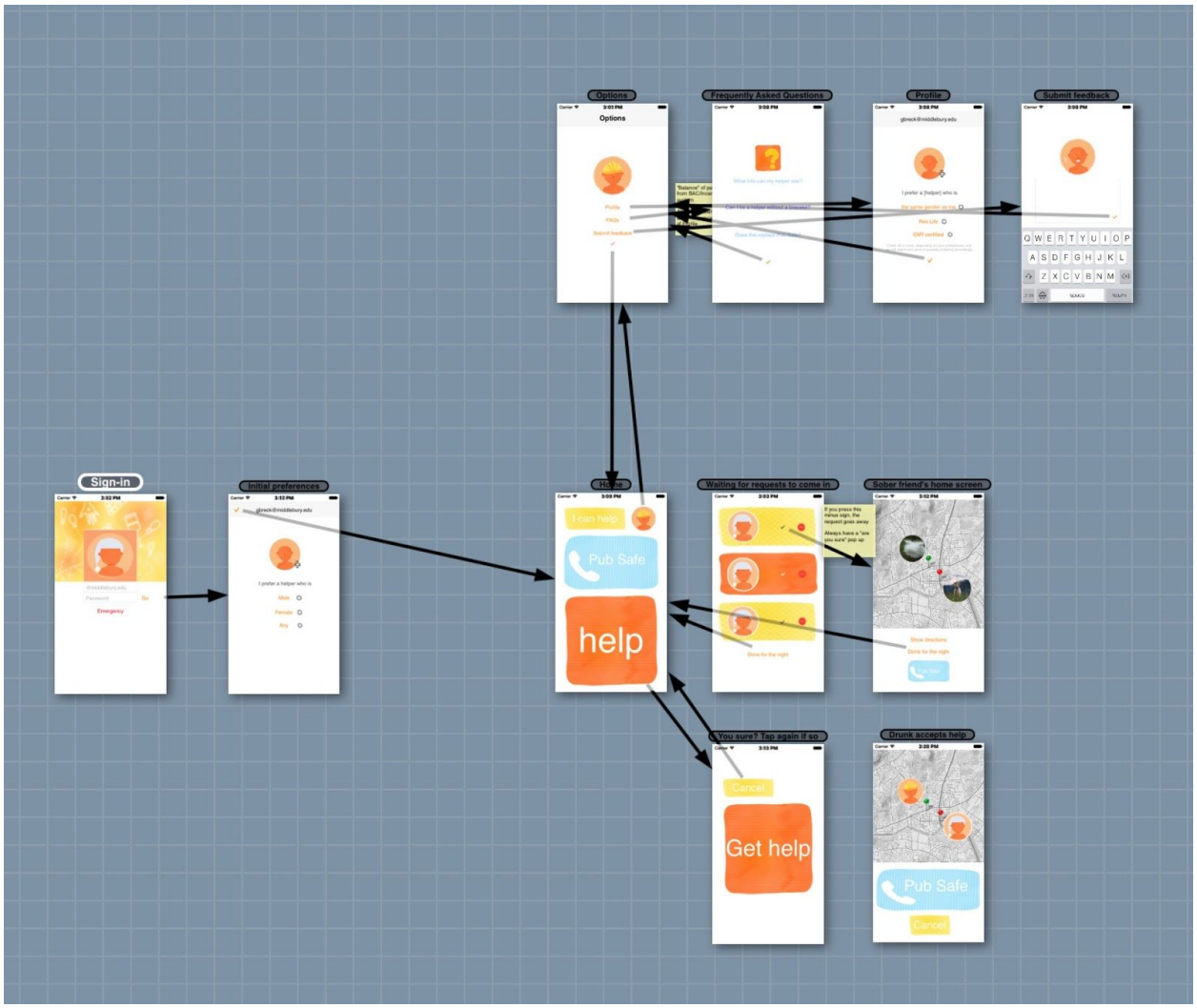
Getting Started

Although not a global phenomenon, the vast majority of Middlebury students use Apple iPhones instead of Android devices. Writing an iOS application struck STEM Innovation 2017 as the most efficient way to enact social change on campus on a wide scale, but Apple's proprietary language and stringent privacy policy welcomed a host of challenges. Swift frameworks are esoteric. Deviating from convention, Apple intends to simplify certain processes concerning type safety and memory management, but in turn complicates seemingly simple processes. Although easy to read, Swift represents a hodgepodge of programming languages such as Java, Python, and Objective-C.

A prerequisite to iOS app development is creating a Developer account in order to download pre release apps on test devices. This step is crucial in the process of debugging since, in certain cases, having applications running on several phones can elucidate select problems. Given our messaging platform and geolocation services destined to serve over 2000 students coupled with the difficulties of event driven programming, downloading the app on several phones more accurately simulates our foreseen audience in the debugging process. With Xcode downloaded and the Developer account set up, programming can begin.

Visual Design

Prior to developing the iOS application's user interface with Xcode, we assembled a mock-up with Groosoft's Blueprint for iPad Pro, which enables a simple drag-and-dropping of iOS elements and their simulated functionalities onto an empty iPhone screen. Here is the resulting user-interface blueprint:



Each graphical element was first hand-drawn with Apple Pencil into Tayasui Sketches and Adobe Draw (two digital-drawing apps for iPad Pro), then exported as transparency-preserving .PNG files for use in Blueprint and eventually Xcode. The same software was used to generate the NightCap banner on the first page of this paper.

User Interface (UI)

- ***What is the UI?***

The user interface (UI) is the front-end component of the application that is displayed to the users.

- ***Purpose of UI design***
  - Improve user experience
  - Graphic Design
  - Typography
  - Navigation

The design of the UI directly influences the user experience. An aesthetically pleasing UI can attract more users through its graphic design, pleasing typography, and flexible navigation.

Figure 1

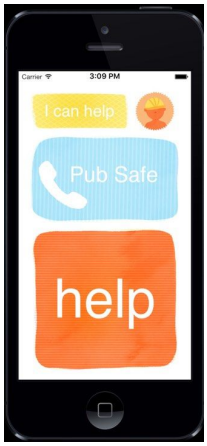


Figure 2

For example, on the HomeViewController (Figure 1), the strategically designed "help" button aims to provide convenience when a user requires assistance.

The initial preference setting will be done when user first log into NightCap. On the HomeViewController, a user can modify his/her own profile setting by tapping on the profile-setting icon, going to profile, and resetting the preferences again.

❖ ***Components of the UI***

- Main.storyboard

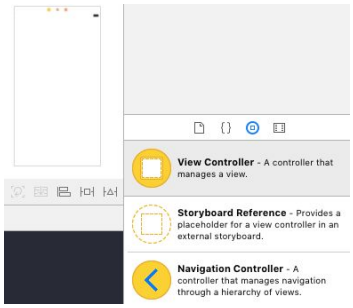
Main.storyboard is a visual representation of the User Interface of an iOS application, a programing canvas for interface designs.

This platform allows the developer to establish a desired sequence of views, which are connected by segue objects performing transitions between this views.

- View Controller

View Controllers are the foundation of the app's internal structure, managing each view in application's view sequence.

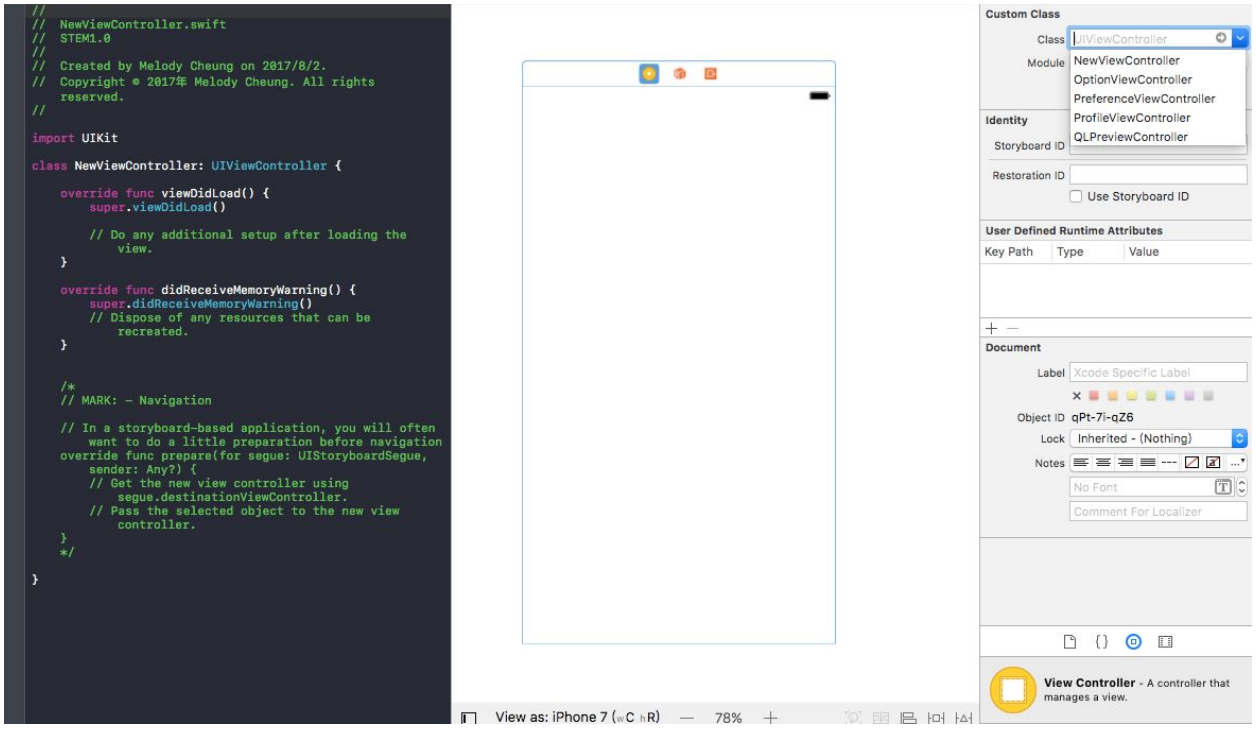
Figure 3



Every new Xcode project has a View Controller generated in Main.storyboard. Adding additional view controllers is as simple as navigating to the Objects Library and dragging it onto the storyboard (Figure 3).

After creating a new View Controller, one must link a new swift file by *clicking on File on the top bar → New→ File→ Cocoa Touch Class→ Select UIViewController in SubClass and name it (Figure 4) → Make sure the language is Swift → Next→ Create.*

Figure 4



This connected Swift file manages all actions taken place within a View Controller. To establish the link between this new swift file and the new view controller, consider the following (Figure 4): *Click on the new view controller→ Show identity inspector → In Custom class, class → Select the swift file just created, in this example, it is "NewViewController".*

- UIButton

A button executes code in response to user interactions. The appearance (Figure 5) is highly customizable. One can change the appearance properties at Attributes inspector.

Figure 5

Figure 6

In order to perform an intended action using a button, connect the button to the code by *Clicking on the button → holding down control → Draw to swift file → Release → Name button → Set connection to "Action" → Click connect (Figure 6)*. In the designated method, the developer can perform desired actions such as performing appropriate segues and sending server requests.

- UILabel

A label is a view that displays non-editable text. Its properties are set in Attributes inspector or the appropriate swift file. Labels, on one hand, can display pre-set static text, but on the other hand, can subject to change.

The sample code is shown in Figure 7.

Figure 7

Hello, m! Welcome to Home Page.

```
override func viewDidAppear(_ animated: Bool) {
    Name.text = "Hello, "+name+"! Welcome to Home Page."
}
```

- UITextField
-

Figure 8

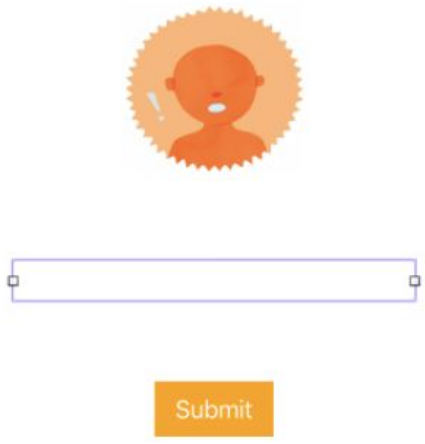


Figure 9

UITextField displays an editable text area which aims to gather user input. In NightCap, the feedback submission page features this UI component(Figure 8).

Most Log-in pages highly rely on text fields in order to allow user to input appropriate login credentials (Figure 9).

In order to protect the password or other classified information, one can select the "secure text entry" box to make text input in a field appear as dots instead of text or numbers.

- UIImageView

Figure 10

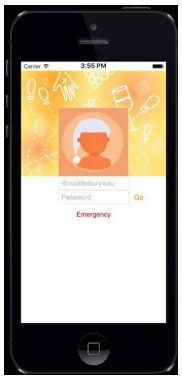


Image view displays a single image, or an animation described by an array of images. It is often used for setting a background image (Figure 10) and providing an aesthetic indicator of the appropriate view.

Simply drag an imageview onto the appropriate view controller and change the properties in Attributes inspector. From the drop-down list in "image", select the desired image and change the scale in content mode.

* All image must be stored in Assets.xcassets folder beforehand.

- UITableView and UITableViewCell

Tableview is a view that uses rows, or TableViewCells, to present data retrieved from a server or database. Cells can be customized by adding a UITableViewCell subclass and connecting it with a Prototype cell.

For example, in NightCap the table view below (Figure 11) will display the username and a general profile picture icon for people who request help. Figure 12 shows how contents are displayed in a prototype cell.

Figure 11

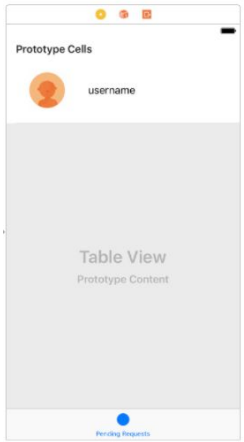


Figure 12

```
//Popolate Table View with some text
public func tableView(_ tableView: UITableView,
    cellForRowAt indexPath: IndexPath) ->
    UITableViewCell
{

    let cell = self.myTableView.dequeueReusableCell
        (withIdentifier: "cell", for: indexPath) as!
        CustomCell

    cell.name.text = helpList[indexPath.row]

    return(cell)
}
```

- Constraints

Adding constraints to UI component in a view helps the app adapt to different screen sizes. They can set positions and sizes for elements in the view controller as well as alignments for a more adaptable design.

- App Icon

The icon of the application requires different sizes to be implemented into different devices. In order to implement the application icon, convert the picture in an online icon converter, download the icon package, go to xcode project Assets.xcassets folder, and drag and drop the right size of picture from icon package into corresponding boxes and the icon will show on home screen.

App Structure

Version 1.0 of Nightcap features an App Delegate class, 12 View Controllers, and a Core class:

We use the App Delegate to perform methods upon both the startup and the shutdown of the application. Executing certain tasks in the background versus the foreground, uploading user information and preferences locally stored in the phone's memory, and initializing a location manager comprise our use of the App Delegate. This portion of the app abstractly represents the glue between the app and the rest of the phone, ultimately managing application use with regard to the plethora of other processes one's phone has running at one time.

A View Controller represents a singular view in the overall hierarchy of views on an app. It controls portions of the user interface and the information flow between the interface and backend of the app. To add a View Controller to a project is as simple as dragging and dropping a View Controller onto the main Storyboard. Segues, whether manual or triggered programmatically, link the views in the overall hierarchical sequence. Xcode allows for developers to drag and drop user interface objects onto views as an outlet connection or action connection. Each View Controller in the project has a file containing Swift code, which manages the actions specific to the View Controller. For example, *HomeViewController.swift* delegates the actions on the home screen while *DisplayList.swift* controls the view containing the helper's list of students needing help at a given time. All of the View Controllers communicate with our *Core.swift* file.

The Core class is the interface between the app's user interface and the server. A static class, which cannot be instantiated, the class represents the central logic hub of the app. First, it stores global variables to keep track of data the entire app requires and enumerations to keep track of state changes we deem important. For example, tracking the view state granted us more flexibility concerning methods we could perform on different views, facilitated by the *setViewState* method. These checks prevent null pointer exceptions and impede crashes of the program. In event-driven programming, the paradigm backing app development, it is imperative to log state changes.

Moreover, Core manages location data. The *serverUpdateLocation* method accurately encompasses our refined location demands specific to Middlebury by accessing the BSSID of the phone's current Wi-Fi router connection coupled with gathering the RSSI of neighboring routers.

Accomplished in *getWifiInformation*, communication to the server about the router to which a phone is connected and the relative signal strengths of neighboring routers paint an unmatched accurate picture of a user's current location. A user's current latitude and longitude are also sent to the server in case a phone is not connected to a Wi-Fi router. First, Core receives latitude and longitude data from the location manager in App Delegate and a conversion to a server-friendly form in the *getCoordinatePair* helper method follows.

Most importantly, Core manages all server requests. When a request is made from any View Controller, *sendServerRequest* is called. In this method, the URL is added to a dictionary that stores all pending requests then receives a request number as a label. A URL Session Data Task is created, which then makes the request to the server. The *serverReceive* method is immediately called following the *sendServerRequest* with a raw string of the HTML content read from the server. In *serverReceive*, the HTML content is parsed by new lines and by tokens, which provide the instructions to follow indicated by the server. Below, we specify a system following the receipt of the server:

> Once the content is split by new lines, based off of the first token in the string, or a tag, send the information following the tag to the appropriate place in the application. For example, the "AK" tag, or an acknowledgement of receipt from the server, delegates segues from view to view based off the view state. Until the "AK" tag is received, no segues can take place. The "MG" tag reads in messages from involved parties and sends it to the appropriate View Controller. For our purposes, from an abstract point of view, the tag is the server's instructions towards the app.

On any mobile platform, a consistent connection to a server is never guaranteed. When using Nightcap, it is very possible that a user attempts to make a request, but it fails to reach the server. To address this concern, we store all requests made to the server in Core and delete them once we receive an acknowledgement of the request from the server. If there are any entries in the dictionary after a specified period of time, each entry is re-sent three more times. Once it completes this loop, a pop-up warning appears on the current View Controller, specified by the view state.

Debugging

With every programming endeavor, there is a host of syntax and logic errors. In fact, most of the time spent programming was sorting out the logic on a blackboard. However, in event-driven programming where there are multiple threads running at once, thread errors are the most predominant type of error and the hardest to track. The order of events in a user session proves to be extremely important and adds an extra dimension of complexity to the problem since foreseeing every course of action is never trivial. The combinations are endless.

To this end, most of the time spent was debugging and defensive programming. That is, we acutely tracked errors on the server logs, programmed pop-ups to indicate app-side errors, and set up protocols previously mentioned (acknowledgements) to prevent thread errors from happening in the future. To assess the quality of the app, we tested the app on several phones simultaneously in an attempt to simulate the real world use of the app. More often than not, in

the debugging process, when several phones use the app at once, it is highly likely that errors will reveal themselves at a higher rate. Thus, singular testing on the simulator has its merits, but does not always suffice. However, when an iPhone experiences a crash or some anomaly, one can plug it into Xcode, reproduce the bug, and read the error logs on the console. All in all, when difficult bugs emerged, the goal was to reproduce the error because only then can you begin to understand the underlying fault in the logic.

Server Structure

The server side of the application was structured in PHP. Information was stored in a centralized directory that contained all the necessary folders and files to execute the PHP code. Server-side code is very different from client-side code in that no functions were created within the PHP files, and instead the whole file was treated as a function that took commands through URL inputs. The server also takes requests rather than having a constantly running code, so actions are initiated by the client/user. When two users are put in communication, the server acts as a hub that both receives and delivers their information. All information is stored/accessed through files (reading, writing, creating), and it is delivered to the app through webpage string reading.

Server Setup

The setup used for the initial testing server was XAMPP development environment (Apache was the only one used). Atom is the suggested editor to write PHP files, though Notepad (or any other generic text editor) works fine. Once the necessary PHP files are placed in the XAMPP "htdocs" subdirectory and the server is running, requests can be made to the server through URL inputs. For example, to call a specific file that takes 2 inputs, the URL input would be:

localhost/example.php?input1=someInput&input2=someInput

This example only works for the device that is running the server. Any requests made extraneously (by any device other than the one the server is being run on) must be done using the server device's IP address. For example:

http://140.233.169.178/example.php?input1=someInput&input2=someInput

These extraneous requests are how the back end of the application communicates with the server.

A very useful device on the server is the error log, which can be found in the "logs" folder of the "apache" directory under XAMPP. This keeps track of any errors that have ever occurred when a request was made. It is very useful since when the app is making requests there is no webpage to view the errors, making the log the easiest way to debug the code. In the same "logs" folder there is also an access log which keeps track of every request that ever reached the

server. This helps decipher if a problem is on the server end or on the back end of the app, because if the request is never logged that means there is a problem with the app (the request never reached the server).

Server and App Communication

It is very important that the people working on the server and the people working on the app work very closely so that there is flawless data flow when the app is running. A suggested method for data flow is Tags, which were used to deliver information to the app once a request was made. One example of a tag is HR, which stands for help request. When the app reads this tag in, it knows to relay to the user that they have a pending help request. One tag used throughout all the PHP files (except *update.php*) was AK, meaning acknowledgment. This defensive protocol was used to ensure that the app didn't segue between screens if the user wasn't connected to the server or if the request didn't go through for other reasons.

Example

1. User clicks "help" button on app
2. App sends request with user's information to *help.php* URL
3. *help.php* does all the necessary file creations to alert sober users of the new request
4. *help.php* responds by echoing an acknowledgment of the request, which lets the app know to segue to the loading screen (waiting for someone to accept)

Main Components

I. BSSID folder
    A. Keeps track of all BSSIDs by having each BSSID as a text file's name and the contents of the text file being the location associated with that BSSID.
II. Help folder
    A. Keeps track of all pending help requests. Its main purpose is to allow sober people who logged in after the request to get the request.
    B. Each text file in this folder is named after the user who made the help request, and the contents in line order are Location, Preference, Request Num.
III. Locations folder
    A. Database of all locations on Middlebury's campus, the files themselves containing two latitude/longitude values to form a boundary around a specified location.
    B. This folder is used only when BSSID information isn't provided or when the given BSSID doesn't match anything in the database. Locations are much more general since it uses lat/long values and won't have any indoor locations listed.
IV. Sober folder

A. Keeps track of all the current users who are logged in as sober (pressed "I can help"). Used to loop through available sober helpers and put help requests inside their folders.

V. *update.php*

A. Location – Takes the user's location and BSSID to have multiple ways to calculate their location. This location is then stored for helpers/helpees to access when put in communication with the user.
B. Help Requests - Looks through the user's help request folder and alerts them to all the available requests.
C. Messages – Looks through the user's messages folder and shows them on screen.
D. Helper/Helpee – When two users are put in communication, one has a helper.txt and the other has a helpee.txt. Through the information in these files, either user can access information about the other.
E. Finishing Sessions
    1. Done – This file assists in finishing sessions by echoing the vital tag "DN Fin" when a done.txt file is found, which lets the app know to segue out of the current session.
    2. Connection – Checks if either user's file hasn't been updated in a while, if so then a connection error is reported.

VI. *help.php*

A. Help Requests – Creates help requests in sober user folders filtered by the specified gender preference. A help request (with preferences saved in the text file) is also created in the global Help folder.

VII. *accept.php*

A. Help Requests – Deletes this help request from all other sober user folders and the global help file
B. Helper/Helpee – Creates a helper.txt file for the requestee and a helping.txt for the sober person. This allows for easy information delivery in *update.php*.
C. Defensive Coding – If someone tries to accept the request once it has been accepted already, there is a CA (cancel accept) tag to let the app know not to segue.

VIII. *regLogin.php*

A. Creates a folder for the user containing:
    1. Messages subdirectory, with a number.txt file inside (contents initialized to 1)
    2. Help request subdirectory
    3. Location text file (loc.txt)
    4. User text file (user.txt)

B. Clears out any extraneous files that may not have been deleted from the last session

6. DrinkTrac: Ethanol Sensor Development

After prompting the student population with questions concerning campus safety, we found that, on the whole, students trust each other. There still exists, however, a level of uncertainty when it comes to mixed drinks. It is inevitable that, during the process of bartending, the drink mixer will produce something with an unknown ethanol concentration. Simply trusting estimates does not suffice. To deliver peace of mind to our users, a method for determining alcohol content accurately had to be developed. Also, students expressed interest in tracking their own level of intoxication. The ethanol sensor we produced is also able to act as a breathalyzer via manipulation of the software script.

Design Philosophy

The original design process started by isolating two non-negotiable requirements: small form-factor and ease of use. To go along with those constraints, design goals were set. Those goals were to provide users with the ability to place a number on the amount of alcohol they are putting into their body, determine their own blood alcohol content quickly and easily, and to have those devices placed in people's hands free of cost. With those constraints in mind, we decided to build the device modularly. In an idealized setting, sensor heads could be swapped out to test for a litany of substances. The STEM 2017 group decided to train our focus on substances that would commonly be utilized at college parties.

Parts Catalogue/Overview:

Arduino Uno - Acting as the base, this unit provides both processing power and electrical energy to the various add-ons necessary for the functioning deliverable. Using the Uno was intuitive and continued refinement was made simple by its easily understandable integrated development environment (IDE). At its simplest, the Arduino microprocessor would be the device collecting an analog reading from the sensor itself and, through various scripts written by the programmer, that analog reading would be interpreted.

Breadboard - these act as a standardized way to connect electronics. By utilizing the positive and negative rails, power can be supplied to individual devices on the board. Once everything is set up, the programmer can decide when, for example, an LED is turned on or off. Each individual piece can, on a breadboard, be wired to the same circuit. Connecting the individual devices together to have them work in tandem was done using this technique.

LED - light emitting diodes. When a suitable voltage is applied, electrons are able to recombine with electron holes within the device, thus release energy as photons. These were used to display certain operating conditions without having to write to the LCD screen.

LCD - liquid crystal display. This device was used to print various states corresponding to ethanol levels in solution. In this application, the screen was used to print 1) whether or not ethanol is present and 2) ethanol concentration.

Adafruit LCD Serial backpack - this add on will save us many pins on the Arduino and make setup more straightforward.

MQ303[1] - the sensing unit. Acting as a variable resistor, the tin-dioxide coating allows for a decrease in electrical resistance when volatile, alcohol-containing compounds come into contact with it. This decrease in resistance increases the voltage. The increase in voltage is measured through the analog zero pin of the Arduino. The heater circuit has to be established and run at 5V for 24 - 48 hours before consistent measurements can be taken.

HC-SR04 - the ultrasonic distance sensing module. In order to record accurate measurements, the sensor had to be within 4 cm of the solution being tested. By tying the measurement protocol to the operation of this device, we ensure that users are given the most descriptive analysis of the substances they choose to test.

Power Supply - the breadboard must be connected to either a 3.7V or 5V power supply at all times. This unit provides power to everything but the Arduino. The Arduino can be powered through either one of its I/O ports via an outlet or a battery.

Manufacturing

Building the device itself took place in two phases: module installation and wiring. Installation was done by first orienting everything on the breadboard to ensure that all of the I/O ports were accessible and that the rangefinder, sensor head, and status LEDs were visible and easily oriented. Although the breadboard streamlined this process and made swapping connections out simple, its form factor would not allow it to be adapted to wearable use. To produce the fully functional prototype:

1. Collect each of the parts listed above along with a 20K ohm resistor (for use with the MQ303 sensor) and 270 ohm resistors (for use with the LEDs, remember - as ohms increase, the LED's brightness will decrease)

2. Solder the necessary components together. Specifically - with the MQ303 sensor - soldering the wires directly to the pins is recommended rather than utilizing a breakout board. The middle pin on each row of three operates the heater. Solder the serial backpack to the LCD unit as described in the startup manual.
3. Wire them in accordance to Figure 1 below. Red wires correspond to power. Black wires correspond to ground. Blue wires correspond to information flow.
4. Insert the following code into the Arduino IDE, compile, and upload it to the board:

```
#define trigPin 13 //Rangefinder
#define echoPin 12 //Rangefinder
#define led 10
#include "Arduino.h"
#if defined(ARDUINO_ARCH_SAMD) || defined(__SAM3X8E__)
  // use pin 18 with Due, pin 1 with Zero or M0 Pro
 #define lcd Serial1
#else
  #include <SoftwareSerial.h>
  SoftwareSerial lcd = SoftwareSerial(0,2);
#endif

int pin8 = 8;
int pin10 = 10;
int sensor = A0;
int sensorValue = 0;


void setup() {
  Serial.begin (9600);
    pinMode(echoPin, INPUT);
    pinMode(trigPin, OUTPUT);
    pinMode(led, OUTPUT);
  pinMode(pin8, OUTPUT);
  lcd.begin(9600);
  lcd.write(0xFE);
  lcd.write(0xD1);
  lcd.write(16);  // 16 columns
  lcd.write(2);   // 2 rows
  delay(10);
  lcd.write(0xFE);
  lcd.write(0x50);
  lcd.write(200);
  delay(10);
  lcd.write(0xFE);
  lcd.write(0x99);
  lcd.write(255);
```

```
  delay(10);
  lcd.write(0xFE);
  lcd.write(0x4B);
  lcd.write(0xFE);
  lcd.write(0x54);
  lcd.write(0xFE);
  lcd.write(0x58);
  delay(10);


}

void loop() {
 long duration, distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(20);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;
  if (distance < 4) {  // This is where the LED On/Off happens
    digitalWrite(led,HIGH);
    sensorValue = analogRead(sensor);
  delay(100);
  // Print out the value you read
  Serial.println(sensorValue, DEC);

  // no alcohol
  if (sensorValue < 200 ) {
    lcd.print("Alcohol Free");

  }

    else {
      lcd.print("Alcohol Present");
            }
    }


  // If sensorValue is greater than 500
  if (sensorValue > 500) {
    // Activate digital output pin 8 - the LED will light up
    digitalWrite(pin8, HIGH);

}
  else {
    digitalWrite(led,LOW);
    digitalWrite(pin8, LOW);
  }

  if (distance >= 200 || distance <= 0) {
    Serial.println("Out of range");
  }
  else {
    Serial.print(distance);
    Serial.println(" cm");
```

```
  }
  delay(500);
}
```

5. By changing the value of the if(sensorValue) method, the programmer can adjust the ethanol concentration at which the sensor trips. Follow this calibration scheme if desired:

	0.01 (% conc.)	1 (% conc.)	5 (% conc.)	10 (% conc.)	20 (% conc.)	100%
Arduino Reading 1	115	220	330	400	486	1023
Arduino Reading 2	150	225	349	397	450	1023
Arduino Reading 3	148	233	341	401	466	1023
Average	137.7	226	340	399.3	467.3	1023
Baseline	79 - 100					
Notes	Readings continually increase as time moves on. The testing protocol should only be active for 10 seconds. Whatever the reading is at that time is the value that will be sent to the back end.	Stepwise operation: have the test run for x seconds. If y number is recorded, display that concentration. For v1, we'll display (yes/no). v2 will have estimate %alc				

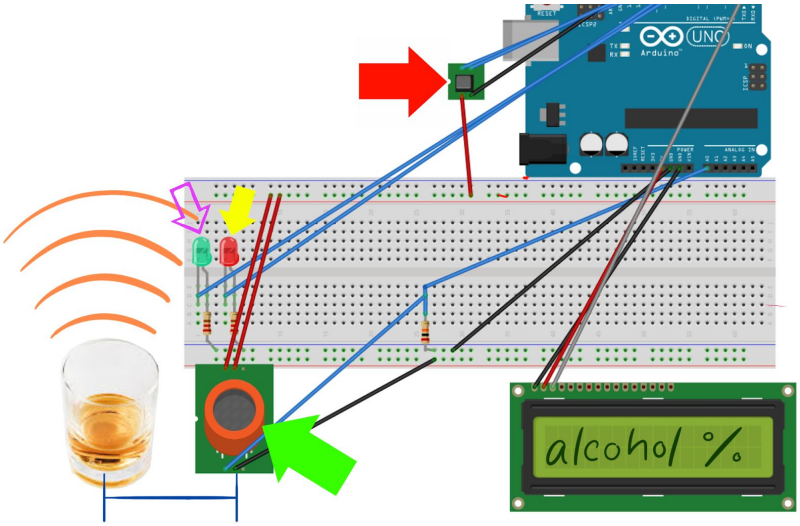


Figure 1. Circuit diagram of the ethanol detector.
Gas-phase ethanol molecules decrease the electrical resistance of the tin dioxide-coated ceramic sensor head[2].The ultrasonic range finder (red arrow) works in tandem with the ethanol detector (green arrow) to ensure that the sensing chamber is within 4 cm, causing the red LED (closed arrow) to blink. The green LED (open arrow) illuminates at a certain % threshold. The LCD screen displays %. Sensitivity range: 0.02 % w/v to 40% w/v ethanol (Circuit diagram created with Fritzing).

Additional Information/Troubleshooting

This section will be used to discuss the hangups and challenges faced during development. Its inclusion will simplify reproduction and iteration.

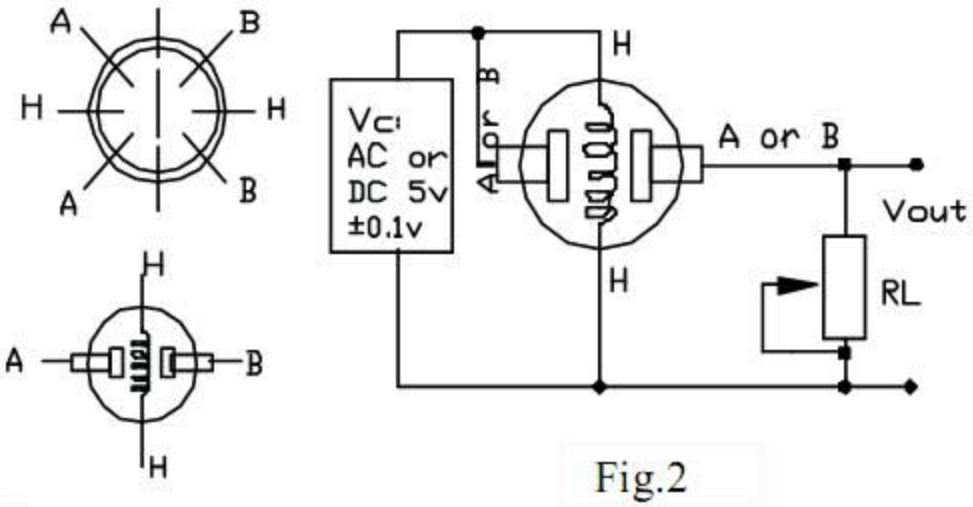


Fig.2

Figure 2: Sensor wiring in-depth.
The setup[3] that was used in the prototype followed this schematic. Although the data sheets show otherwise, we found this to be the optimal configuration. The Vout is connected to the Arduino analog 0 pin. The sensor needs to be heated to operate consistently. Power should be drawn from an external power source rather than the Arduino. When using a battery, a transistor or logic-level mosfet can be used to switch it on and off.

Continued Development

Avenues for further iteration on the hardware side are plentiful. The modularity of the design allows for all manner of detection-based technology to be swapped in. Additional hardware will be created following student interest. As the NightCap program grows, we will track user-logged feedback to highlight areas in need of further safety consciousness.

Feedback has already kickstarted the development of new and improved hardware. During the summer symposium, audience members prompted our thinking by mentioning stories of tourists being harmed by tainted alcohol[4]. The compound causing harm, methanol, is a byproduct of improper alcoholic beverage manufacturing. Preliminary testing has shown that the sensor is capable of detecting the presence of methanol in solution. To flesh-out its methanol sensing capabilities, experiments must be run to isolate differences in the sensor response curve when ethanol and methanol are separate, combined, and in dilute quantities.

7. Final Model

STEM has produced an iOS app that streamlines communication between NightCap Assistants and student users, and anticipates further development of devices centered on safety consciousness. Both the app and the sensor accompany a full program proposal that bridges two existing Middlebury communities: the peer-to-peer support group network (ResLife, MiddSafe, GreenDot, etc.) and the Department of Public Safety. As a whole, Nightcap facilitates peer support in nonurgent situations, filling a niche in our community's safety network.

NightCap Program Description

NightCap will be a student-run program that facilitates peer-to-peer assistance by employing students to act as assistive resources for students on weekends in non-emergent situations. A student in duress will be able to make a request for assistance using the smartphone application, while a NightCap assistant will receive the request with their GPS location on a campus map. The NightCap program NightCap assistants would be responsible for going to the student's location upon request and staying with them until the student is in good company or until the assistant and the student agree that the student is fit to be by themselves. NightCap

assistants will be trained to identify situations that require Public Safety to intervene and would be required to do so when necessary. NightCap assistants would not replace or interfere with any existing services and would ideally work in tandem with systems already in place. NightCap assistants could work in close communication with Public Safety in a capacity that has yet to be determined.

General Protocol

After an assistant arrives at the student requester's location, the student would assess the student for signs of alcohol poisoning. In the event that warning signs for alcohol poisoning exist, the assistant would immediately call Public Safety. The student would also use skills learned from training to evaluate the need for Public Safety and contact them if need be. A checklist would be completed and would be relayed back to Public Safety upon completion.The assistant would fulfill their responsibility to assist the student and inform the College about the incident through a written report.

The NightCap assistant's responsibility would be to stay with the student requester as long as necessary until both the assistant and the requester agreed that requester was sufficiently well to be by him or her self, or until the requester was in safe company. At the end of the request the student would send back a report to Public Safety with details of the request. The report would not include identifying information.

The Role of the Public Safety Department

In unsafe or escalating situations, NightCap assistants would inform Public Safety. Decision over which situations would and would not require Public Safety would be agreed upon by NightCap, Public Safety, Middlebury Risk Management and Middlebury Legal Counsel. Special Circumstances would require Public Safety to be contacted for intervention. This includes but is not limited to: (1) signs of alcohol poisoning, (2) signs of sexual assault, (3) violent or belligerent behavior, and (4) request of the student. Circumstances that NightCap assistants would not call Public Safety would mainly include mild intoxication. Specific circumstances that require and do not require Public Safety will be established ahead of time.

Trainings

NightCap assistants would receive any and all necessary trainings to make sure they could safely decide if a situation was dangerous or not and whether or not they would need to call Public Safety. The specific trainings would be decided and approved by Public Safety, Risk Management, and the Department of Health and Wellness. Important topics to cover in trainings

include: 1) Recognizing signs of alcohol poisoning, 2) Recognizing signs of sexual assault, 3) Responding to violent or belligerent behavior. The following required trainings in our proposal of the program are as follows:

- TIPS Training
- Bystander Intervention Training
- Think About It: Alcohol, Drug, and Sexual Violence Training
- Party Host Workshop
- Hazing Training, Bystander Action Training

Smartphone Application

The Nightcap application would make the program much more accessible. It would also simplify data collection and could include other safety features for the students. The basic features are as follows:

1. Request a NightCap assistant for yourself or someone else.
2. Call Public Safety or 911.
3. Share geolocation with a NightCap assistant, and also include a message. The application would be automated to share students geolocation at regular time intervals (eg. every 30 minutes).

Funding

Funding for a pilot program would be supported by the STEM Innovation project. The estimated cost for the pilot program is $4,000. For details see “Pilot Proposal for NightCap Program”. Whether the students are employees or volunteers has not yet been decided by the college. If these students were to be employed by the school, we recommend that they be paid at the specialist rate.

NightCap Program Proposal

Student-run NightCap facilitates peer-to-peer assistance by employing students as assistive resources in nonurgent situations. Should such circumstances escalate, assistants seek Public Safety’s intervention. Such conditions (to be agreed upon by NightCap, Public Safety, Middlebury Risk Management and Middlebury Legal Counsel) include but are not limited to signs of alcohol poisoning, sexual assault, and violent or belligerent behavior.

Circumstances not requiring intervention include underage drinking, mild intoxication, or non-escalating inebriation. The student’s condition will be re-evaluated by the Assistant every 10-15 minutes until the end of the session. Ideally, NightCap Assistants will available to respond

to requests from the hours of 9PM to 3AM from Thursday to Sunday, during which students can request assistance using the smartphone app, while at least two on-call assistants will respond accordingly.

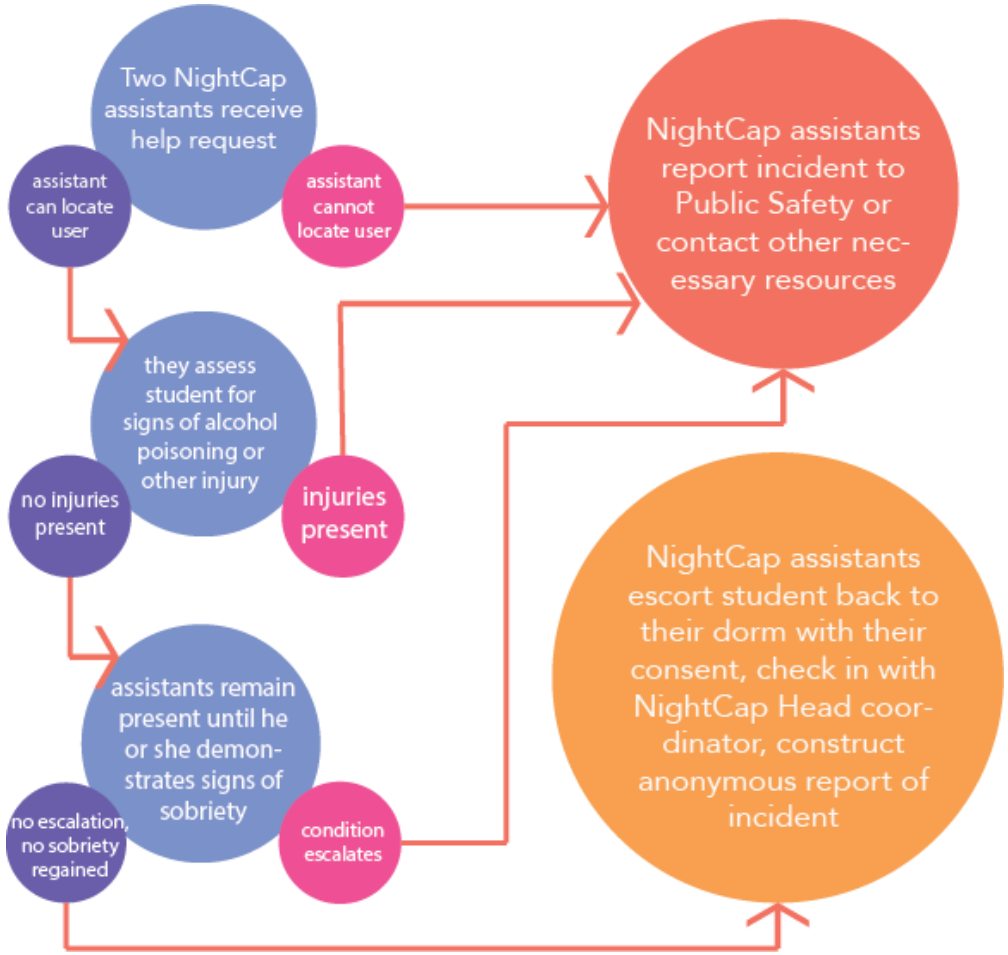


Figure 1. Envisioned Protocol for the NightCap Assistant.

NightCap Assistants will be held accountable for fulfilling the following conditions ***to the best of their ability***:

1. Prevent their assigned student in duress from consuming any more alcohol or recreational drugs
2. Comply with protocols taught in relevant trainings
3. Be present with the assigned student until the student demonstrates the ability to care for themselves
4. Not allow the assigned student to operate an automobile or take part in other dangerous acts
5. Watch for signs of alcohol poisoning and alert Public Safety when necessary
6. Call relevant campus resources in any escalating emergency or ambiguous situation

NightCap Assistants will be "on call" for their time. They will not be permitted to consume alcohol or any other recreational drugs while working.

Protocol for Special Circumstances:

1. If special circumstances prevent a NightCap assistant from arriving to their predetermined destination within 15 minutes of a request, then the request will be canceled and the user will be put into contact with Public Safety.
2. Any request for assistance will be accepted by two NightCap assistants. If one assistant is late or missing from a session, then the other assistant will be allowed to provide support, but must alert Public Safety before doing so.
3. NightCap assistants are not mandatory reporters or confidential resources by law. Therefore, any personal violation or other observed emergency is to be handled as a Middlebury student.
4. Meeting places will be established outside of campus buildings as NightCap assistants will not be granted special access to campus spaces. However, if a student user requests the NightCap assistant to enter a suite, room, or personal space via direct message, then it will be considered permission to enter that space. If the student user changes their mind and wishes for the NightCap student to leave a space prematurely, then the assistant will comply to those wishes but call Public Safety and report the incident.
5. In the event that a student user requests help for an non-consensual third party, the NightCap assistant will be required to call Public Safety in order to ensure the safety of the student in duress.
6. NightCap assistants will not respond to requests made off-campus or respond to requests made for non-students.

<u>NightCap Program Terms of Service</u>

*These Terms of Service constitute an agreement between you and the NightCap program (as defined below) governing your access to and use of, including any subdomains thereof, and any other domain through which NightCap makes the Services available, our smart device application, and application program interfaces (collectively, "**Application**") and all associated services (collectively, "**NightCap Services**")*

The services are provided “as is” and “as available”. Nightcap makes no representation, warranty, or guarantee regarding the reliability, timeliness, quality, suitability, or availability of the services or any services or goods requested through the use of the services, or that the services will be uninterrupted or error-free. NightCap does not guarantee the quality, suitability, safety or ability of its assistants. You agree that the entire risk arising out of your use of the services, and any service or good requested in connection therewith, remains solely with you, to the maximum extent permitted under applicable law.

NightCap shall not be liable for indirect, incidental, special, exemplary, punitive, or consequential damages, including personal injury or property damage related to, in connection

with, or otherwise resulting from any use of the services, regardless of the negligence (either active, affirmative, sole, or concurrent) of NightCap.

NightCap shall not be liable for any damages, liability or losses arising out of your use of or reliance on the services or your inability to access or use the services. NightCap shall not be liable for delay or failure in performance resulting from causes beyond a NightCap advocate's reasonable control. You acknowledge that NightCap providing peer support services may not be professionally licensed or permitted.

The limitations and disclaimer in this section do not purport to limit liability or alter your rights as a consumer that cannot be excluded under applicable law. You agree to indemnify and hold NightCap and its affiliates and their representatives harmless from any and all claims, demands, losses, liabilities, and expenses (including attorneys' fees), arising out of or in connection with (i) your use of the Services; (ii) your break or violation of any of these Terms; (iii) NightCap's use of your User Content; or (iv) your violation of the rights of any third party.

NightCap Program assistants shall not be liable for any accidents due to alcohol poisoning, sexual assault, or violent behavior as long as the advocate is performing their obligations to the best of their ability. NightCap assistants shall not be granted special access to campus buildings or student spaces without a formal request. By requesting the service from the NightCap Program, you are granting permission for the assistant to access the involved space.

You agree that some assistant services may carry inherent risk, and by participating in those Services, you choose to assume those risks voluntarily. For example, some assistant Services may carry risk of illness, bodily injury, disability, or death, and you freely and willfully assume those risks by choosing to participate in those advocate Services. You assume full responsibility for the choices you make before, during and after your participation in an assistant Service.

If you are bringing a minor as an additional guest, you are solely responsible for the supervision of that minor throughout the duration of your assistant Service and to the maximum extent permitted by law, you agree to release and hold harmless NightCap from all liabilities and claims that arise in any way from any injury, death, loss or harm that occurs to that minor during the advocate Service or in any way related to your Advocate Service.

Privacy Policy

This Privacy Policy explains how we collect, use, process, and disclose your information across the NightCap Platform. By agreeing to the Terms and Conditions of our program, you also agree to the following Privacy Policy.

1. **Information We Collect**
    a. Information You Give Us
        i. **Account Information.** When you register your smartphone using the NightSafe Application, we require certain information such as your name, gender, and preferences for your assistant host
        ii. **Profile Information.** You have the option to provide certain information for ease of use. For example, you will be able to submit the names of certain assistants that you may be prohibited from interacting with.
        iii. **Communications with Nightcap and other Members.** When you communicate with NightCap or use the platform to communicate with its assistants, we collect information about your communication and any information you choose to provide for the duration of that session.
        iv. **Other Information.** You may otherwise choose to provide information when you fill in a feedback form, update information through your account, respond to surveys, post to community forums, participate in promotions, or use other features of the NightCap Platform.
        v. **Information from NightCap Advocates.** This might include any complaints or concerns raised by the NightSafe assistants for the students' or assistants' safety.
    b. Information We Automatically Collect from your Use of the NightCap Smartphone Application
        i. Location. When you use certain features of the NightCap Platform, we collect your geolocation. Most mobile devices will allow you to control or disable the use of geolocation services for application in the device's settings menu.
        ii. Log Data. We automatically collect log information when you use the NightCap application. That information includes, among other things: details about how you've used the application (including connections to NightSafe assistants) access times, hardware and software information, or device information.

How We Use Information We Collect

c. Provide, Improve, and Develop the Platform
    i. Enable you to access and use the NightCap application
    ii. Enable you to communicate with NightCap assistants
    iii. Provide customer service by responding to complaints and concerns for the program

d. Create and Maintain a Safe Environment

i. Comply with our legal obligations
ii. Conduct checks against databases and other information sources
iii. Conduct investigations and risk assessments
iv. Resolve any disputes with any of our users and enforce our agreements with NightCap Advocates
v. Enforce our Terms of Service and other policies

2. Sharing and Disclosing

a. Safety and Compliance with Law

i. NightCap may disclose your information to courts, law enforcement or governmental authorities, if and to the extent we are required to do so by law or if such disclosure is reasonably necessary: (i) to comply with legal process and to respond to claims asserted against NightCap, (ii) to respond to verified requests relating to a criminal investigation or alleged or suspected illegal activity or any other activity that may expose us, you, or any other participant to legal liability, (iiI) to enforce and administer our Terms of Service or other agreements with Members, (iv) for fraud investigation and prevention, risk assessment, customer support, product development and debugging purposes, or (v) to protect the rights, property or personal safety of NightCap, its advocates, and its users.

b. Sharing between Advocates and Student Users.

i. To help facilitate a connection between advocates and student users, we share your information with the advocates as follows:
    1. When you submit a request for an advocate, certain information is shared that you have provided to the NightCap member, including your full name and any other information you agree to share. When the request is confirmed and a session opens, your location will also be disclosed.
    2. The identity of the NightCap advocate will be disclosed to the student user as soon as the request is accepted. If the student is unwilling to accept help from a NightCap advocate, then they will have the opportunity to cancel the request.
    3. While NightCap Advocates are not confidential resources, they will be expected to perform their obligations with discretion. However, they will not be responsible for breaking intended means of confidentiality when looking after another student.

3. Security

a. We are continuously implementing and updating administrative, technical, and physical security measures to help protect your information against unauthorized

access, loss, destruction, or alteration. However, the Internet is not a 100% secure environment so we can't guarantee the security of the transmission or storage of your information.

4. **Contact**
    a. If you have any questions or complaints about this Privacy Policy or NightCap's information handling practices, you may contact us at:
        i. dplayer@middlebury.edu
        ii. jbowllan@middlebury.edu
        iii. eguerra@middlebury.edu
        iv. jravishankar@middlebury.edu
        v. jsilverstein@middlebury.edu

Pilot Proposal for NightCap

Objective: To fully implement this proposal, STEM Innovation aims to demonstrate the utility and effectiveness of our proposed program in practice. Our team will demonstrate that Middlebury students will participate by volunteering as assistants and by requesting the support service. The effectiveness of our program will be demonstrated by closely monitoring hospital visits, user activity, and any relevant data collected for accidents on campus. We will promote the program on campus and measure student approval and interest in participating by advertising go/nightcap, social media, and positive feedback.

Budget: Total (To be sponsored by STEM Innovation): $4,000.00

- **Application server**: 250$ for one year
- **Training for first wave of assistants**: 1,000$ for one year
- **Promotions and Marketing Budget**: 600$
    - Posters
    - Promotions
        - Get 3 friends to register the app and get 5$ in Grille Money
    - Other
- **Private contractors**: $2,150
    - Five students will be hired as private contractors and will be on call for 6 hours each weekend night (Saturday and Sunday) from October to May and be compensated approximately 10$ per hour.

Obligations of the STEM group:
At least one member of the STEM Innovation team will have completed the following trainings and will run monthly meetings with the 5 private contractors for the 7 months in action:

- TIPS Training

- Bystander Intervention Training
- Think About It: Alcohol, Drug, and Sexual Violence Training
- Party Host Workshop
- Hazing Training, Bystander Action Training

Software Updates:
John Anthony Bowllan and Josh Ravishankar will update the NightCap smartphone application and troubleshoot when needed.

Check-In:
At the conclusion for each request for support, the private contractor will check in physically with the STEM Innovation representative and will construct a brief report of the incident that will be available to the College for review.

Timeline (2017-2018)

- **August**
    - The STEM Innovation team will complete the NightCap smartphone application and ethanol sensing device. We will present a completed proposal to to the College Council and Middlebury's Risk Management Team and make any revisions necessary.

- **September**
    - The list of private contractors to participate in the pilot program will be finalized and will complete necessary trainings to be active in October.

- **October**
    - The program will officially become active to current ResLife staff on campus on an "as available" basis
    - While the application will be open to anyone who wants to download it and request service, it will be formally proposed to the above groups for use. Each member interested will sign any necessary Terms and Conditions to gain access to the program services.

- **November- June**
    - At least one meeting will occur per month with at least one representative of STEM Innovation, in addition to Ravishankar or Bowllan and the NightCap Assistants. These meetings will be open to any College Administrators that might wish to attend. The following will be addressed at each meeting:
        - Any problems or concerns with the software/program design
        - An overview of user activity

- Any special circumstances
- Constructive criticism voiced by students and filed complaints
- Morale

Measures of Efficacy

We are hoping to see a reduction in problematic behavior associated with heavy drinking (dorm damage, sexual assault, disrespect for staff, etc.).

8. Future Work

- **College-granted approval for NightCap Pilot Program:** receive permission to implement consolidated program plan
- **Demonstrate utility:** verify that students volunteer as assistants and request support; collect hospital-visit frequency, user activity, and incident details
- **Full Implementation at Middlebury College:** promote on campus via go/nightcap and social media; gauge student approval through feedback
- **Expand to Multi-Purpose Platform:** aid in/for off-campus events, student visitors, and suicide prevention
- **Expand to other college campuses:** fulfil potential to help students beyond Middlebury

9. Acknowledgements

This program was made possible by generous funding from the Hearst Family and the President's Office.

Special thanks to Amy Rose and Beth Eliason for budgeting and shipping, and also to the faculty and staff that guided the progress of this project: Dean of Students Baishakhi Taylor, Director of Business Services Matthew Curran, Associate Dean of Students Douglas Adams, Executive Director for Parton Health and Wellness Gus Jordan, Professor of Psychology Suzanne Gurland, Associate Director of Public Safety Dan Giotti, Director of Public Safety Liza Burchard, Public Safety Officer Paul, Director of Health and Wellness Barbara McCall, Benj Deppman of Deppman Law PLC, and Officer Rick Garey of the Essex County Police Department.

10. References

[1]Banzi, Massimo. Getting started with Arduino. Sebastopol, California: OReilly, 2015. Print.

[2] B. P. J. de Lacy Costello, Sensors and Actuators B: Chemical Volume 87, Issue 1, 2002

[3] Chengxiang Wang, Sensors 2010, 10(3), 2088-2106

[4] "Methanol poisoning warning to travellers." NHS Choices. NHS, n.d. Web. 02 Aug. 2017.

[5]Middlebury Campus. November 12th, 2009. Sober friends take the place of duty office. Retrieved from https://middleburycampus.com/248/news/sober-friends-take-place-of-duty-office/
[6] Middlebury College. n.d.. Alcohol and Other Drugs. Retrieved from http://www.middlebury.edu/about/handbook/student_policies/alcohol_drugs_policy
[7]Middlebury College. May 4th, 2012. Task Force on Alcohol and Social Life: Final Report. Retrieved from https://middfiles.middlebury.edu/middinfo/Dean_Of_The_College/Task%20Force%20on%20Alcohol%20and%20Social%20Life%20Report%202012.pdf
[8] "Storyboard," Apple Developer, accessed August 2, 2017, https://developer.apple.com/library/content/documentation/General/Conceptual/Devpedia-CocoaApp/Storyboard.html