COMPANY: C-WOLF

WHISPER

ADVANCED SOFTWARE DESIGN PROJECT

DEVELOPERS:
DAVID SMITH
JACQUES BEAUVOIR
AARON PINEDA

ABSTRACT:

C*Wolf-Whisper* is an iOS application that acts as a communication channel for Sonoma State students only. The message board and instant chat will serve as a platform to communicate about various topics: homework, classes, jokes, and general entertainment. To ensure that there will only be Sonoma State students accessing the application, we will implement a username/password system using the Sonoma State email as a unique form of identification. The architecture for the project will utilize the MVC paradigm. The project was initially planned to use a relational database to illustrate the connection between objects within the models, but decided to use FireBase which is a NoSQL database. These models will transfer information to the views as a way to represent the data for users to alter. As the users click and type, the application will either store information into the database or switch to a different view. The MVC pattern will allow us to manage each tier of CWolf-Whisper with relative ease. The goal of CWolf-Whisper is to allow students to connect via iPhone and to allow the developers to build a practical application.

GOALS:

· Achieve a deeper understanding of Swift and its libraries
· Create a deliverable product that can be deployed in the app store for students
· Have a user-friendly communication service for students
· Create an application that can be easily maintained, scaled to meet high traffic demands, and expanded by future developers

PLATFORMS:

The application will be exclusively for iPhone, programmed in Swift, with backend elements done in Google's Firebase. Firebase is a very powerful development platform that will help us with creating JSON style elements and databases.

Features:

1. Forum
2. PageViewController
    a. Forum TableViewController
    b. Message Board for users
    c. Category View Controller lets user narrow forums down to interests
3. Category View
    a. Add New Forum
    b. Home (resets the forums to home settings)
    c. History - not implemented
    d. App info a quick page about us
    e. Logout - causes segfault atm - not implemented
4. Instant Chat - We call the chats Whispers
5. About the App

Initially, we were going to implement a home screen with 8 buttons: Forum, Meme of the Day, CWolf Insight, Instant Message, Used Textbooks, Map, About the App, FAQ. Rather than have a home screen we decided to take the user straight to the forum page. We believe taking the user straight to the forums page will create a sleeker, more compact navigation experience for the user. From the forum page the user can swipe left or right in order to access the 2 other views, Instant Chat and Categories. Below is an outline of challenges, successes, and functionality corresponding to different sections of the application.

Database :

The project's database was developed with Google's Firebase, a mobile and web application development platform which gives us the capability of storing and syncing data across all clients in real time. Input is stored as JSON and can be accessed on various platforms. In the future, if we decide to create an Android: Whisper application, we can have users share the database. To access the database, we have a struct with static variables set to different paths within the database - the navigation is very similar to a tree:

```
static let databaseRoot = Database.database().reference()
static let databaseChats = databaseRoot.child("chats")
static let databaseForums = databaseRoot.child("Forums")
static let databaseUsers = databaseRoot.child("Users")
```

In order to fetch data from the database we set-up a path and observe one of the structures (chats,Forums,Users). We continue by accessing and manipulating a snapshot value, which holds information from the structure:

```
Database.database().reference().child("Forums").observe(.childAdded, with: { (snapshot) in
    if let dictionary = snapshot.value as? [String: AnyObject] {
        let forum = Forum(forum: dictionary)
        self.forum.append(forum)
```

We'll explain how data is put into the database in the proceeding "Instant Chat" section.

Instant Chat :

The Instant chat was created with the intent to be similar to the Youtube Live chat structure, where multiple people are capable of logging in to communicate with everyone on the application. So how did we fair implementing this idea? Currently, we have the capability of multiple users logging into our application and positing to the chat. Messages sent by the user produce a blue bubble, while incoming messages from all other users appear in green. There are a few features within the chat that we did not have time to implement, which are profile pictures next to the chat bubble, the username displayed above the bubble appears randomly, and attachments. Next we will talk about how the chat was implemented.

With regards to functionality, we utilized an open source library called JSQMessages created by Jesse Squires, which was installed via CocoaPods. The library allows us to use a

JSQMessage View Controller, which creates a view similar to the iphone messaging system and provides methods to input user messages into a bubble:

```
lazy var outgoingBubble: JSQMessagesBubbleImage = { // BUBBLE FUNCTION 1
    return JSQMessagesBubbleImageFactory()!.outgoingMessagesBubbleImage(with:
        UIColor.jsq_messageBubbleBlue())
}()

lazy var incomingBubble: JSQMessagesBubbleImage = { // BUBBLE FUNCTION 2
    return JSQMessagesBubbleImageFactory()!.incomingMessagesBubbleImage(with:
        UIColor.jsq_messageBubbleGreen())
}()
```

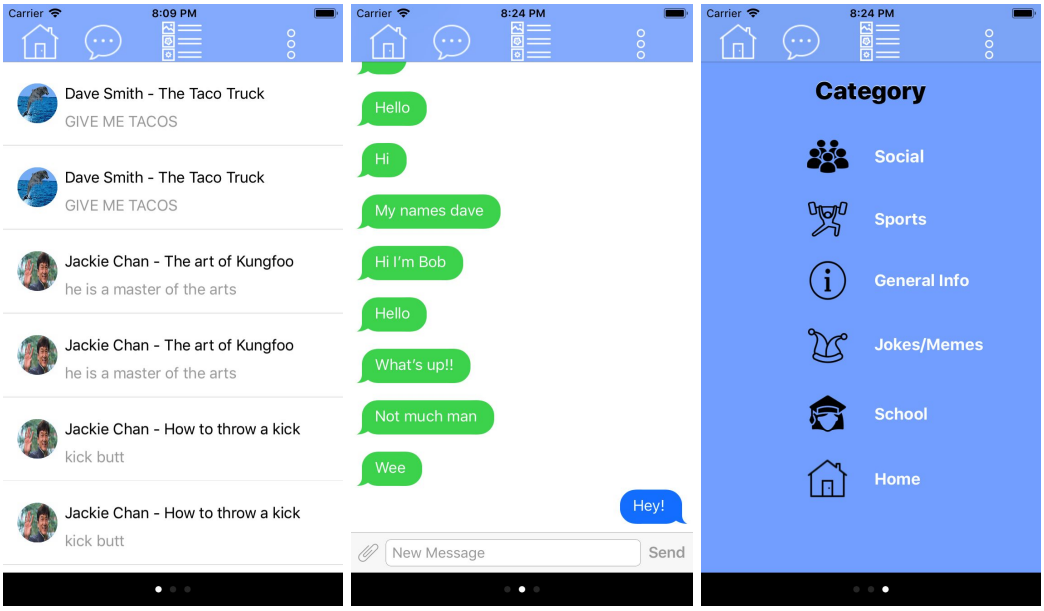
The data collected from user input is injected to our database in a dictionary format where it is subsequently fetched to populate the bubbles. The insertion follows a reference path that allows us to traverse our collection tree. In the code below we have a “ref” variable created that is set to a path within our database with an auto create method. The database automatically creates new children everytime a new message is entered.

```
let ref = Constants.refs.databaseChats.childByAutoId()
let message = ["sender_id": senderId, "name": senderDisplayName, "text": text]
ref.setValue(message)
```

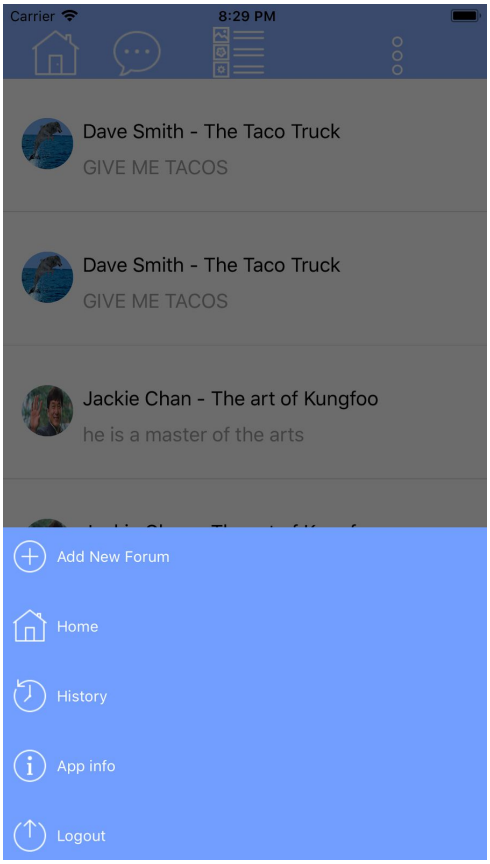
Each message has a unique id associated with it generated by a randomization function choosing a number between 0 and 999,999. We set this value to a string because we had issues fetching integer values from the database.

Interface / flow of program :

Started with just a user and Login viewControllers which quickly grew into a more complex flow of control. If I had known how much cleaner programing without the Storyboard is, I would have done everything programmatically. However are project can be clearly followed looking at our Storyboard. most of are segues are done with Storyboard ID and a select few are overridden. The users get the option to log in or register. After they have done this they may login and will be presented with are rootViewController which is a pageViewController. It has three options The forum page, a chat page where they can chat with other students, and a category page that changes there forum page to the desired choice.



As you can notice all three have a Menu button thats animates up and gives the following option



The Add New Forum segues to a viewcontroller that allows you to add a forum to our database that everyone can see. Home reloads all the home data (we can choose that). History is not

implemented yet but the idea was each user would have an array of forums they have liked, bookmarked, or have commented on. So history would return all of these forums so the user can easily can access previous things they are interested in. We also wanted to add a hot topic option where the top 4 of each forum type with a rating being of liked/commented/rated would make the hot topic results and would refresh monthly. The app info button takes the user to a contact us page and tells them a bit about us. The logout button is not implemented yet and will segfault or doesnt work.

The forums if clicked take you to the details of that forum. And at the bottom there is a button to like the forum and a button to take you to comments of the forum. In the future we would like to have a bookmark, that would append the forum id to that user's history array/ bookmark array and store it back into the database. The like button I thought was working and sending and updating in the database but it seems bugged at the moment. The number of comments is not implemented since comment section is also not implemented.
The overall user interface looks professional and hopefully in the future will have even more things to improve the user's experience.

Back End / Implementation Details :

Initially we were fetching data from a URL which was displayed from my blue public server. It was initially set up to grab the JSON objects from the URL create a observer and when it finished downloading would populate the table view Controller that way. But because of Database setup problems we changed to grab the data differently. The data was grabbed from the firebase database with a fetching function and each element was appended to a array of Forums. Using this array we use setters and getters in are class to access and store data locally and setup Labels, Views, Images, etc. Filters were applied before the appending the data if we wanted only certain JSONS. For example are Forum objects all had a forum_type and if we wanted social all we had to do was filter as we were appending them to are array of data. The TableViewController was made of up custome made cells from ForumCell class that used Constraints and a customized function to help set constraints found in the Extensions of the project. Each cell will use at index path and then set there data using the Forum class getters and

setters with are forum array of JSON objects. I had a bug that took a while to understand which was I was creating cells on top of already made cells. I needed to implement a function called prepareForReuse:

```
override func prepareForReuse() {
    super.prepareForReuse()
    //ForumPicture.image = nil
    subTitleLabelToSet.text = ""
    TitleLabelToSet.text = ""
    ProfilePictureToSet.image = nil
}
```

This fixed the error of overlapping titles and cells since if you scrolled through the previous cell labels and pictures never got reset and would cause some ugly appearances.

We used a pageViewController which helps you slide between viewcontrollers without using segues or the storyboard (besides storyboard id). This class set up an array of Viewcontrollers which when slid would take you to the index of the next or previous page (depending on which way the user dragged). It was setup to go in a circle so if you go past its bounds it will send you to the back or front respectively. In the future We would like to add more ViewControllers to the pageViewController. Another feature i would like to implement is highlight the icons/buttons in the navigation bar that belong to that viewController that the user is at. This would give a better flow and understanding of where the user is in the app. I tried to do this but it was difficult to get the current index the pageViewController was at. For some reason when it swiped it would change my index twice. It called its page swipe twice I tried it in many spots but with no luck. I hope to revisit this problem in the future. See below how it was

initially setup

```
lazy var VCarray: [UIViewController] = {
    return [self.VCinstance(name: "HomeViewController" ),
            self.VCinstance(name: "MessageViewController" ),
            self.VCinstance(name: "CatagoryViewController" )] //These are the storyboard IDs that will
                shown on swiping
                                                              //change name to which vieewwcontroller you
}()
var viewController: [UIViewController] = []

 func VCinstance(name: String) -> UIViewController {
    return UIStoryboard(name: "Main", bundle: nil).instantiateViewController(withIdentifier: name)
}

override func viewDidLoad() {
    super.viewDidLoad()
    //JUST added

    if let navController = self.navigationController {
        viewController = navController.viewControllers as [UIViewController]
    //do changes etc
    }
    UINavigationBar.appearance().titleTextAttributes = [NSAttributedStringKey.foregroundColor:UIColor.w

    self.dataSource = self
    self.delegate = self
    //homeNavButton.tintColor = UIColor.black
    if let firstVC = VCarray.first {
        setViewControllers([firstVC], direction: .forward, animated: true, completion: nil)
    }
```

The menu button is a custom class setup to create an animation of a sliding out menu. It is comprised of collectionViews similar to table views which dequeueReusableCell and didSelectItemAt indexpath: it also had a func to return the height and a func to return count. I implemented a highlighted function that changes the color of the items in the cell to give an appearance of being highlighted similar to how the tableView was done. I created to functions to one to reveal the menu and one to dismiss it. Below is the functions where reveal sets a view of dark grey animates a sliding up of a collection view and the dismissal gets called if they clicked outside the collection view otherwise it goes to did select row at indexpath and after its finished its animation back up it goes to the correct segue associated with that cell. The menu button nested into the Root navigation Controller which initially caused a ton of bugs since i was not setting up the navigation controllers correctly.

```
func revealMenu() {
    //show menu
    let screenSize = UIScreen.main.bounds
    let screenWidth = screenSize.width
    let screenHeight = screenSize.height


    if let window = UIApplication.shared.keyWindow{

        blackView.backgroundColor = UIColor(white: 0, alpha: 0.5)

        blackView.addGestureRecognizer(UITapGestureRecognizer(target: self, action:
            #selector(handleDismiss)))

        window.addSubview(blackView)
        window.addSubview(collectionView)
        window.backgroundColor = UIColor.rgb(red: 131, green: 177, blue: 255)
        let height: CGFloat = CGFloat(settings.count * 50)
        blackView.frame = window.frame
        blackView.alpha = 0
        let CGrect = CGRect(origin: CGPoint(x: 0,y :window.frame.height), size: CGSize(width: 500, height:
            screenHeight))
        let CGrect2 = CGRect(origin: CGPoint(x: 0,y : 450), size: CGSize(width: 500, height: screenHeight))

        collectionView.frame = CGrect

        UIView.animate(withDuration: 1.0, delay: 0, usingSpringWithDamping: 1, initialSpringVelocity: 1,
            options: .curveEaseOut, animations: { self.blackView.alpha = 1
            self.collectionView.frame = CGrect2
        }, completion: nil)

    }
}

@objc func handleDismiss(setting: Setting) {
    UIView.animate(withDuration: 0.5, delay: 0, usingSpringWithDamping: 1, initialSpringVelocity: 1, options:
        .curveEaseOut, animations: {

        self.blackView.alpha = 0

        if let window = UIApplication.shared.keyWindow {
            let CGrect3 = CGRect(origin: CGPoint(x: 0,y :window.frame.height), size: CGSize(width:
                self.collectionView.frame.width, height: self.collectionView.frame.height))
            self.collectionView.frame = CGrect3
        }

    })

    { (completed: Bool) in
        print("comepleted now trying to call function")
        var tempString: String?
        tempString = setting.name

        if tempString != "" {
                self.forumPageVC?.whichSettingButtonWasClicked(stringToCheck: tempString!)
        }
        else{}
    }
}
```

Start Project / Run It

1. Open root folder and then open the project via workspace and not .project
2. Drag in our provided google-service.plist because the file is corrupted after being transferred. delete the previous one since it is empty and will cause issues.
3. Run the app with iphone 8 or 7 emulator
4. To use multiple emulators run one then change emulator by the play button and press run again for both. This way you can test the chat.
5. Post login -> swipe left and right to navigate if you hit a category you must swipe back to forum section for the filter to apply.
6. In the top right there is a menu button click to display and choose accordingly (logout and history -> not implemented).