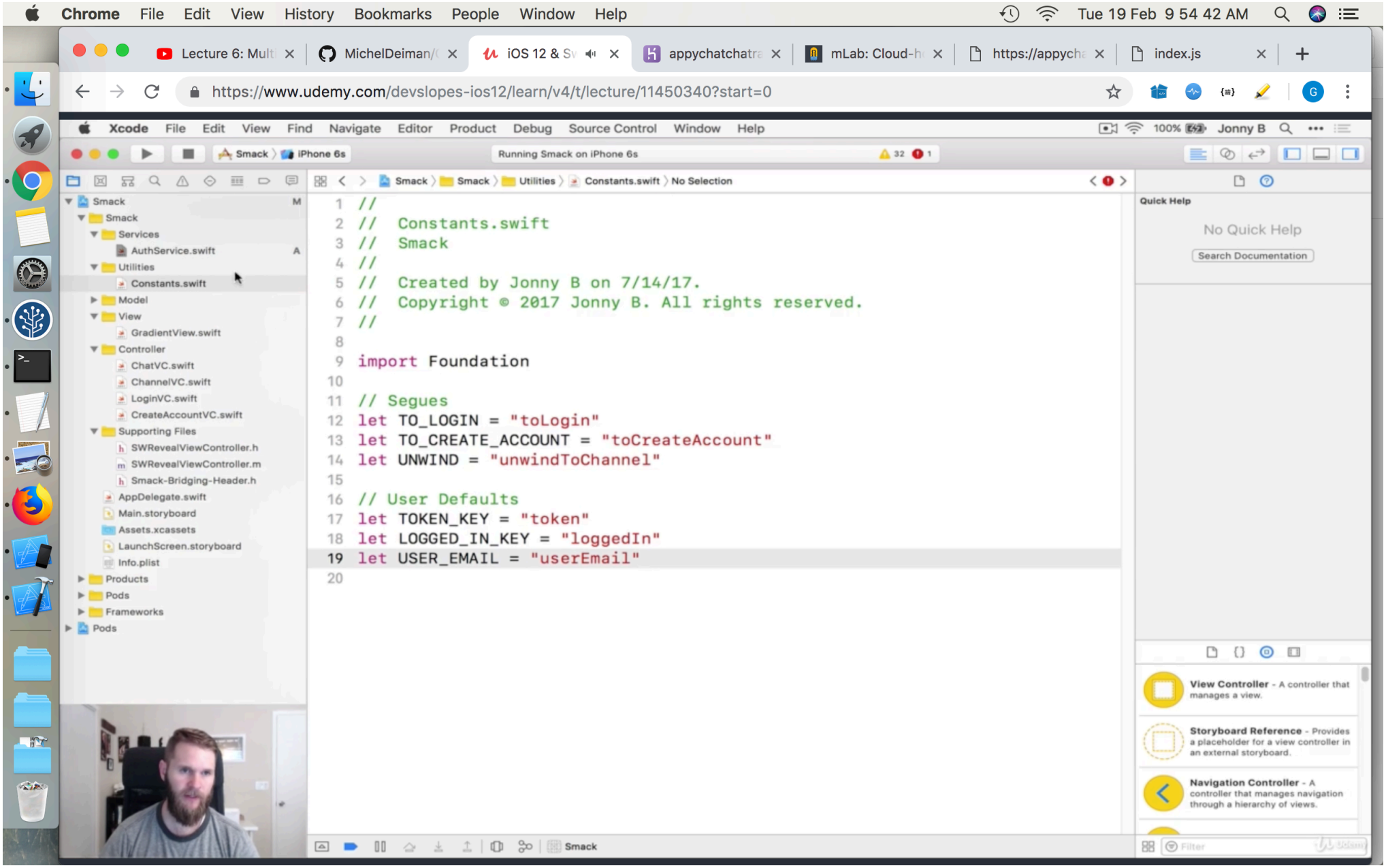
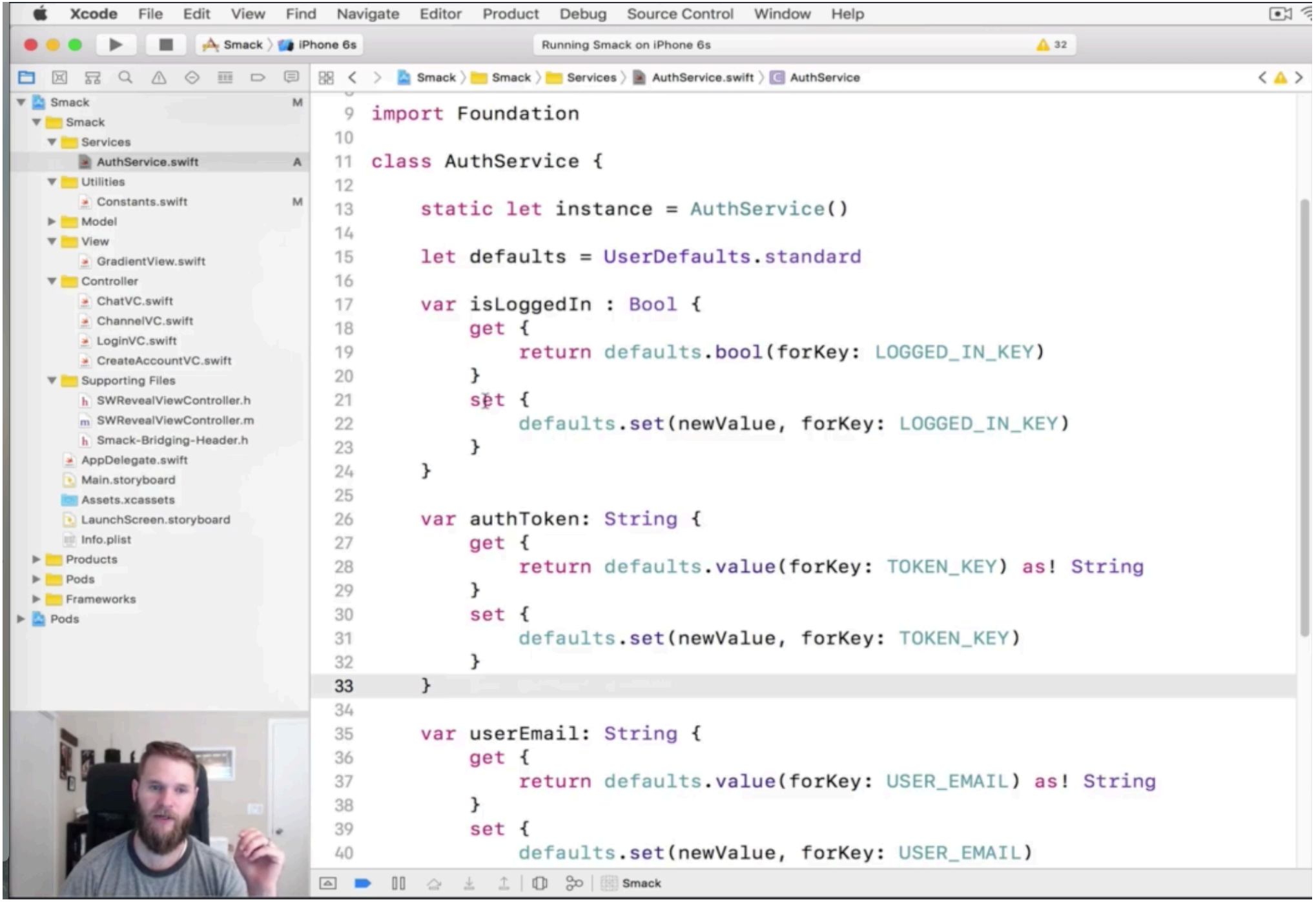
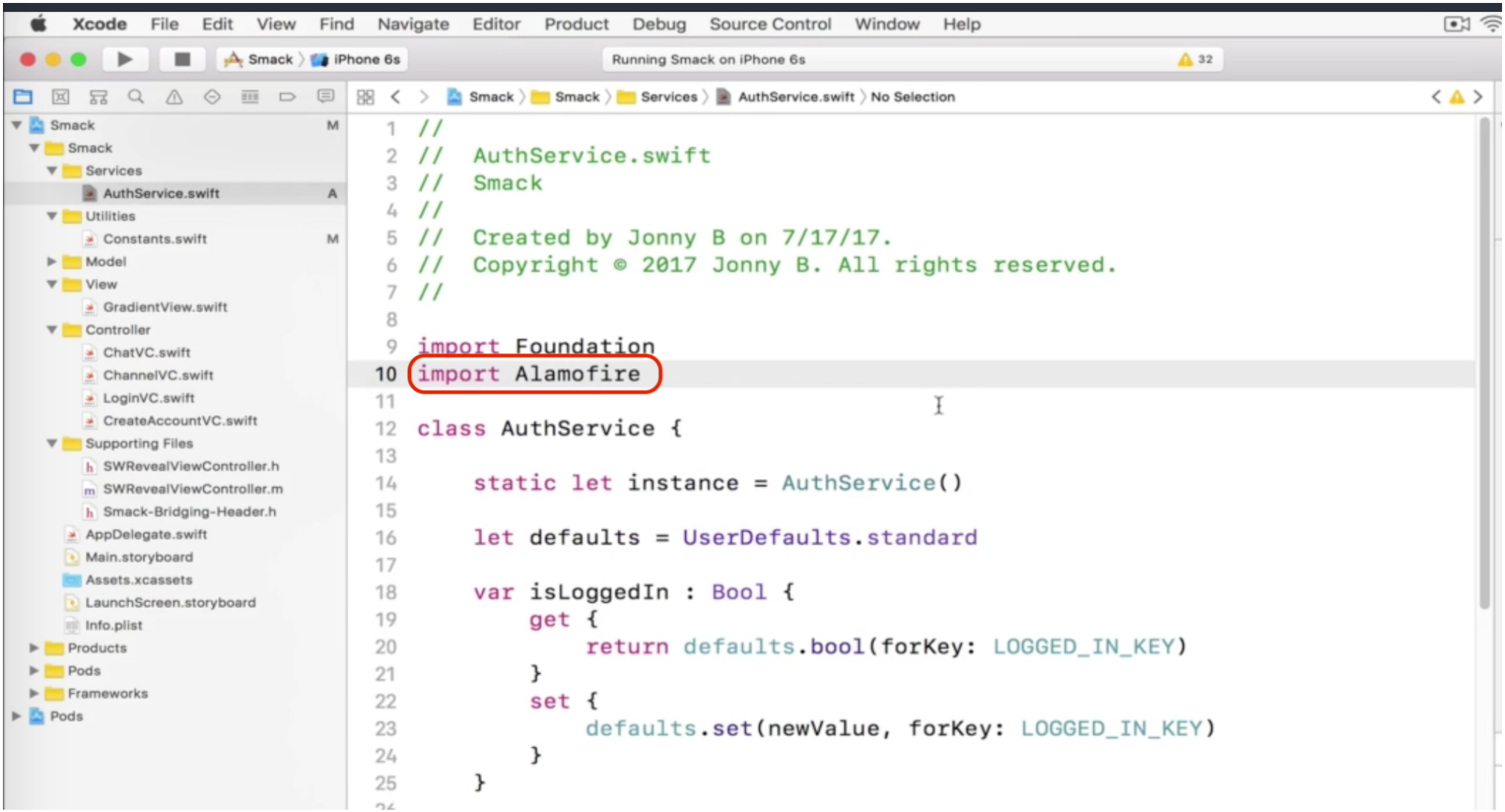

Chrome File Edit View History Bookmarks People Window Help
Tue 19 Feb 9 54 42 AM
Lecture 6: Mult
MichelDeiman/(
iOS 12 & Sv
appychatchatra
mLab: Cloud-h
https://appycha
index.js
https://www.udemy.com/devslopes-ios12/learn/v4/t/lecture/11450340?start=0
Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Jonny B
Smack iPhone 6s
Running Smack on iPhone 6s
Smack Smack Utilities Constants.swift No Selection
Smack
Services
AuthService.swift
Utilities
Constants.swift
Model
View
GradientView.swift
Controller
ChatVC.swift
ChannelVC.swift
LoginVC.swift
CreateAccountVC.swift
Supporting Files
SWRevealViewController.h
SWRevealViewController.m
Smack-Bridging-Header.h
AppDelegate.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
Pods
Frameworks
Pods
1 //
2 // Constants.swift
3 // Smack
4 //
5 // Created by Jonny B on 7/14/17.
6 // Copyright © 2017 Jonny B. All rights reserved.
7 //
8
9 import Foundation
10
11 // Segues
12 let TO_LOGIN = "toLogin"
13 let TO_CREATE_ACCOUNT = "toCreateAccount"
14 let UNWIND = "unwindToChannel"
15
16 // User Defaults
17 let TOKEN_KEY = "token"
18 let LOGGED_IN_KEY = "loggedIn"
19 let USER_EMAIL = "userEmail"
20
Quick Help
No Quick Help
Search Documentation
View Controller - A controller that manages a view.
Storyboard Reference - Provides a placeholder for a view controller in an external storyboard.
Navigation Controller - A controller that manages navigation through a hierarchy of views.
Smack


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Smack iPhone 6s
Running Smack on iPhone 6s
Smack Smack Services AuthService.swift AuthService
Smack
Services
AuthService.swift
Utilities
Constants.swift
Model
View
GradientView.swift
Controller
ChatVC.swift
ChannelVC.swift
LoginVC.swift
CreateAccountVC.swift
Supporting Files
SWRevealViewController.h
SWRevealViewController.m
Smack-Bridging-Header.h
AppDelegate.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
Pods
Frameworks
Pods
import Foundation
class AuthService {
static let instance = AuthService()
let defaults = UserDefaults.standard
var isLoggedIn : Bool {
get {
return defaults.bool(forKey: LOGGED_IN_KEY)
}
set {
defaults.set(newValue, forKey: LOGGED_IN_KEY)
}
}
var authToken: String {
get {
return defaults.value(forKey: TOKEN_KEY) as! String
}
set {
defaults.set(newValue, forKey: TOKEN_KEY)
}
}
var userEmail: String {
get {
return defaults.value(forKey: USER_EMAIL) as! String
}
set {
defaults.set(newValue, forKey: USER_EMAIL)
Smack


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Smack iPhone 6s
Running Smack on iPhone 6s
Smack Smack Services AuthService.swift No Selection
Smack
Smack
Services
AuthService.swift
Utilities
Constants.swift
Model
View
GradientView.swift
Controller
ChatVC.swift
ChannelVC.swift
LoginVC.swift
CreateAccountVC.swift
Supporting Files
SWRevealViewController.h
SWRevealViewController.m
Smack-Bridging-Header.h
AppDelegate.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
Pods
Frameworks
Pods
//
// AuthService.swift
// Smack
//
// Created by Jonny B on 7/17/17.
// Copyright © 2017 Jonny B. All rights reserved.
//
import Foundation
import Alamofire
class AuthService {
static let instance = AuthService()
let defaults = UserDefaults.standard
var isLoggedIn : Bool {
get {
return defaults.bool(forKey: LOGGED_IN_KEY)
}
set {
defaults.set(newValue, forKey: LOGGED_IN_KEY)
}
}

Resources available

will be listed as either `Alamofire iOS`, `Alamofire macOS`, `Alamofire tvOS` or `Alamofire watchOS`.

- And that's it!

> The `Alamofire.framework` is automagically added as a target dependency, linked framework and embedded framework in a copy files build phase which is all you need to build on the simulator and a device.

Usage

Alomafire

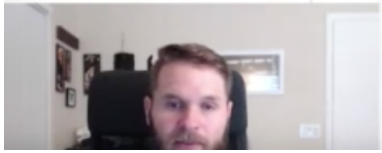
Making a Request

```
import Alamofire

Alamofire.request("https://httpbin.org/get")
```

Response Handling

Handling the `Response` of a `Request` made in Alamofire involves chaining a response handler onto the `Request`.

```
Alamofire.request("https://httpbin.org/get").responseJSON { response in
    print("Request: \(String(describing: response.request))")   // original url request
    print("Response: \(String(describing: response.response))") // http url response
    print("Result: \(response.result)")                         // response serialization result

    if let json = response.result.value {
        print("JSON: \(json)") // serialized json response
    }

    if let data = response.data, let utf8Text = String(data: data, encoding: .utf8) {
        print("Data: \(utf8Text)") // original server data as UTF8 string
    }
```



Response Data Handler

The `responseData` handler uses the `responseDataSerializer` (the object that serializes the server data into some other type) to extract the `Data` returned by the server. If no errors occur and `Data` is returned, the response `Result` will be a `.success` and the `value` will be of type `Data`.

```
Alamofire.request("https://httpbin.org/get").responseData { response in
    debugPrint("All Response Info: \(response)")

    if let data = response.result.value, let utf8Text = String(data: data, encoding: .utf8) {
        print("Data: \(utf8Text)")
    }
}
```

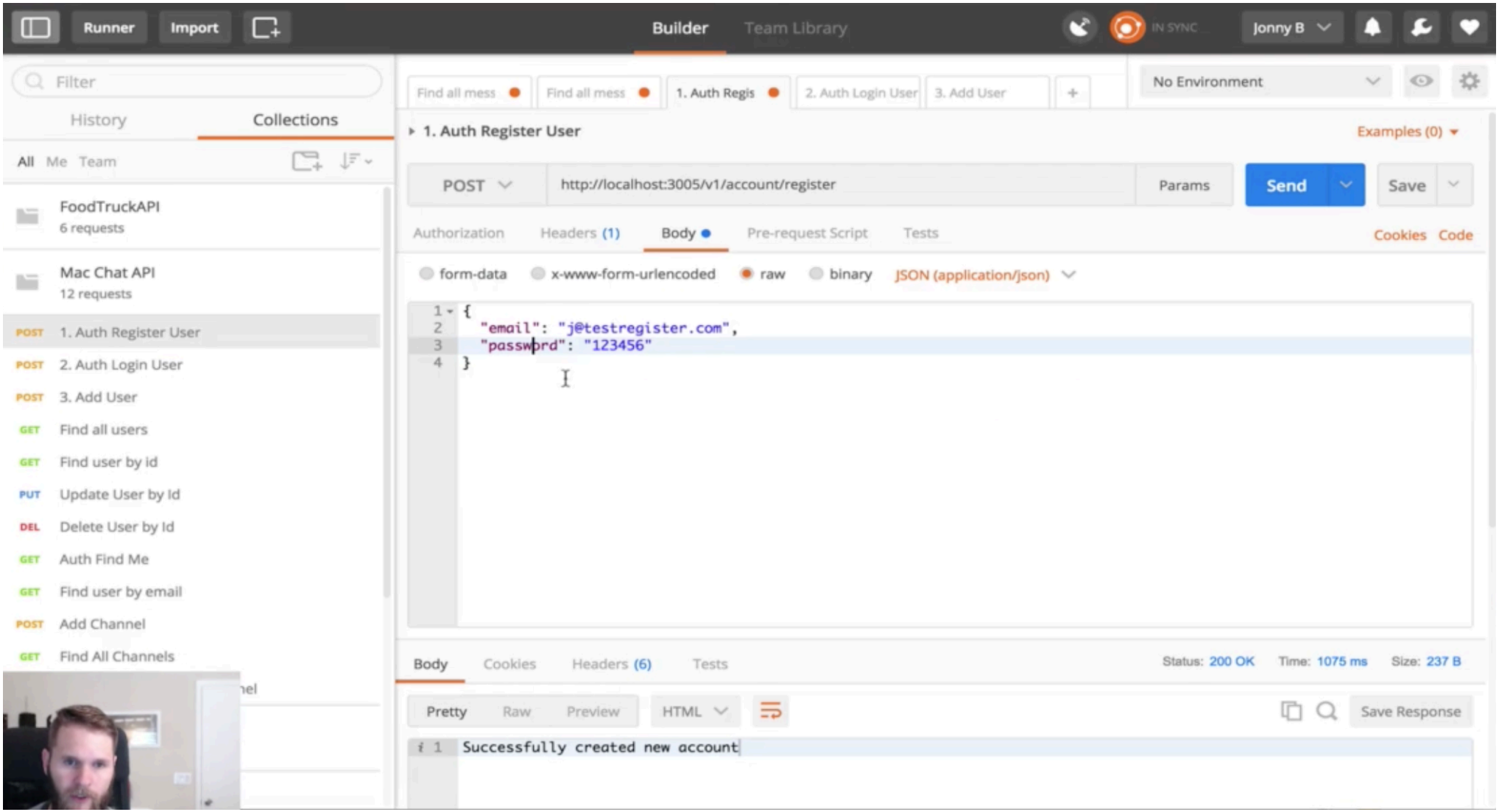
Response String Handler

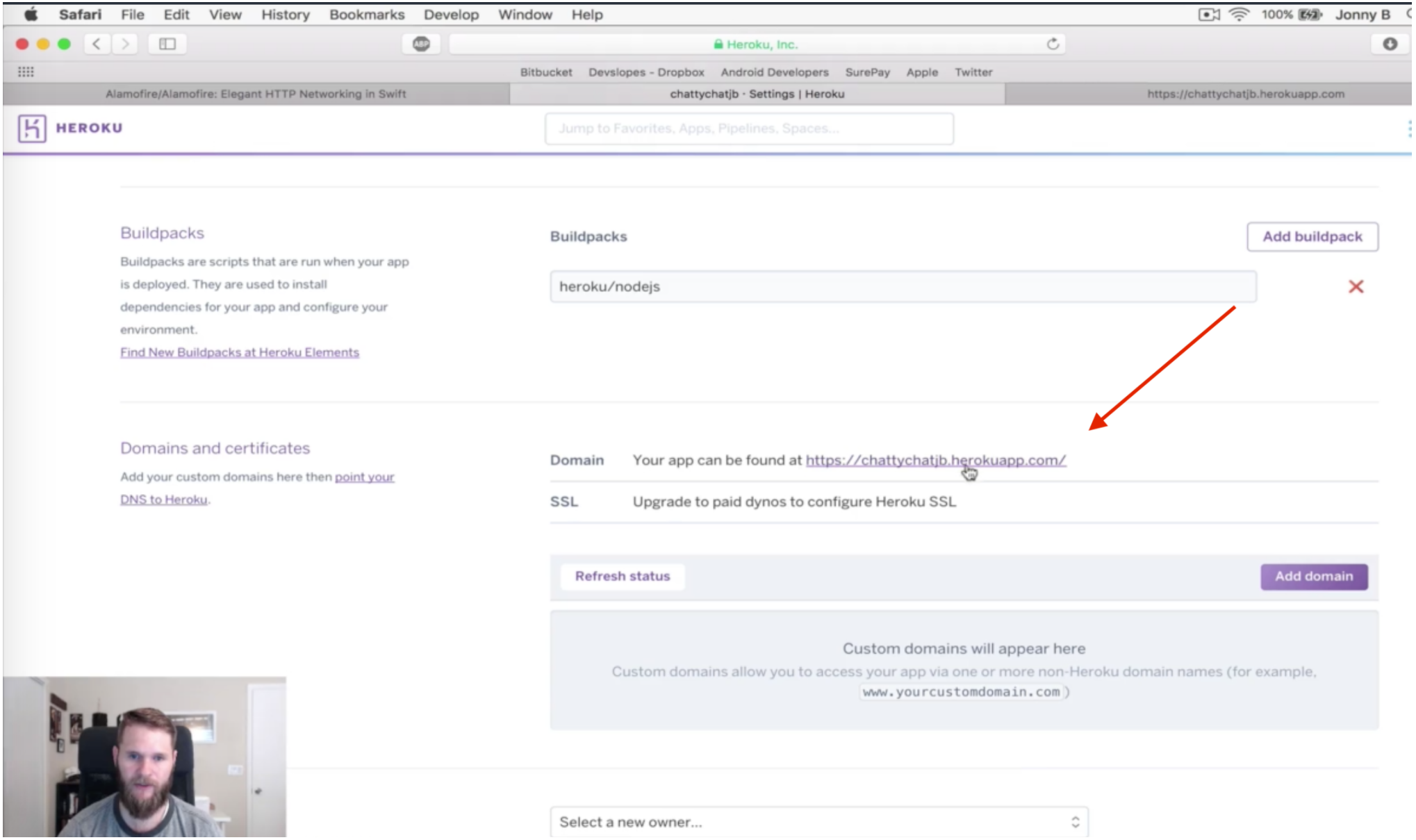
The `responseString` handler uses the `responseStringSerializer` to convert the `Data` returned by the server into a `String` with the specified encoding. If no errors occur and the server data is successfully serialized into a `String`, the response `Result` will be a `.success` and the `value` will be of type `String`.

```
Alamofire.request("https://httpbin.org/get").responseString { response in
    print("Success: \(response.result.isSuccess)")
    print("Response String: \(response.result.value)")
}
```

> If no encoding is specified, Alamofire will use the text encoding specified in the `HTTPURLResponse` from the server. If the text encoding cannot be determined by the server response, it defaults to `.isoLatin1`.

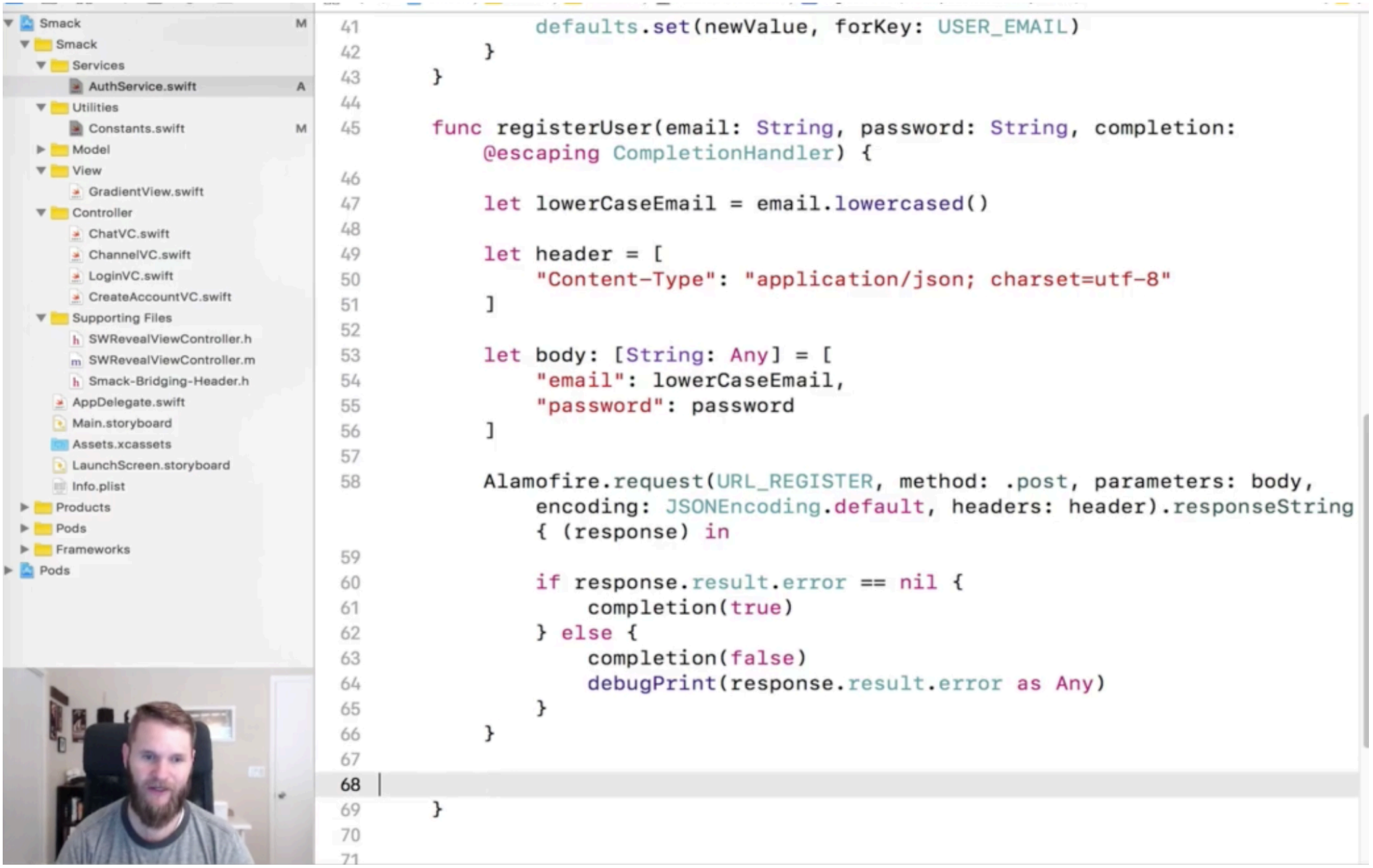
Response JSON Handler


Runner
Import
Builder
Team Library
IN SYNC
Jonny B
Filter
History
Collections
All Me Team
FoodTruckAPI
6 requests
Mac Chat API
12 requests
POST 1. Auth Register User
POST 2. Auth Login User
POST 3. Add User
GET Find all users
GET Find user by id
PUT Update User by Id
DEL Delete User by Id
GET Auth Find Me
GET Find user by email
POST Add Channel
GET Find All Channels
Find all mess
Find all mess
1. Auth Regis
2. Auth Login User
3. Add User
No Environment
1. Auth Register User
Examples (0)
POST
http://localhost:3005/v1/account/register
Params
Send
Save
Authorization
Headers (1)
Body
Pre-request Script
Tests
Cookies
Code
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
{
"email": "j@testregister.com",
"password": "123456"
}
Body
Cookies
Headers (6)
Tests
Status: 200 OK
Time: 1075 ms
Size: 237 B
Pretty
Raw
Preview
HTML
Save Response
Successfully created new account


Safari File Edit View History Bookmarks Develop Window Help
100% Jonny B
Heroku, Inc.
Bitbucket Devslopes - Dropbox Android Developers SurePay Apple Twitter
Alamofire/Alamofire: Elegant HTTP Networking in Swift
chattychatjb · Settings | Heroku
https://chattychatjb.herokuapp.com
HEROKU
Jump to Favorites, Apps, Pipelines, Spaces...
Buildpacks
Buildpacks are scripts that are run when your app is deployed. They are used to install dependencies for your app and configure your environment.
Find New Buildpacks at Heroku Elements
Buildpacks
Add buildpack
heroku/nodejs
Domains and certificates
Add your custom domains here then point your DNS to Heroku.
Domain
Your app can be found at https://chattychatjb.herokuapp.com/
SSL
Upgrade to paid dynos to configure Heroku SSL
Refresh status
Add domain
Custom domains will appear here
Custom domains allow you to access your app via one or more non-Heroku domain names (for example, www.yourcustomdomain.com)
Select a new owner...

Smack > Smack > Utilities > Constants.swift > No Selection

```swift
//
//  Constants.swift
//  Smack
//
//  Created by Jonny B on 7/14/17.
//  Copyright © 2017 Jonny B. All rights reserved.
//

import Foundation

typealias CompletionHandler = (_ Success: Bool) -> ()

// URL Constants
let BASE_URL = "https://chattychatjb.herokuapp.com/v1/"
let URL_REGISTER = "\(BASE_URL)account/register"

// Segues
let TO_LOGIN = "toLogin"
let TO_CREATE_ACCOUNT = "toCreateAccount"
let UNWIND = "unwindToChannel"

// User Defaults
let TOKEN_KEY = "token"
let LOGGED_IN_KEY = "loggedIn"
let USER_EMAIL = "userEmail"

```


Smack
Smack
Services
AuthService.swift
Utilities
Constants.swift
Model
View
GradientView.swift
Controller
ChatVC.swift
ChannelVC.swift
LoginVC.swift
CreateAccountVC.swift
Supporting Files
SWRevealViewController.h
SWRevealViewController.m
Smack-Bridging-Header.h
AppDelegate.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
Pods
Frameworks
Pods
defaults.set(newValue, forKey: USER_EMAIL)
}
}
func registerUser(email: String, password: String, completion: @escaping CompletionHandler) {
let lowerCaseEmail = email.lowercased()
let header = [
"Content-Type": "application/json; charset=utf-8"
]
let body: [String: Any] = [
"email": lowerCaseEmail,
"password": password
]
Alamofire.request(URL_REGISTER, method: .post, parameters: body, encoding: JSONEncoding.default, headers: header).responseString { (response) in
if response.result.error == nil {
completion(true)
} else {
completion(false)
debugPrint(response.result.error as Any)
}
}
}