

# IMPLEMENTACION

Gestión de Proyectos de software



Instituto Tecnológico de la Laguna

18131215	Gustavo Maximiliano Ambriz Zamarripa
18131395	Christian Emmanuel Escalera Cerda

## Hambuerguer view controller

```
1 import UIKit
2
3 protocol HamburgerViewControllerDelegate {
4     func hideHamburgerMenu()
5 }
6 class HamburgerViewController: UIViewController {
7
8     var delegate : HamburgerViewControllerDelegate?
9
10
11     @IBOutlet weak var profilePictureImage: UIImageView!
12
13     @IBOutlet weak var mainBackgroundView: UIView!
14
15     override func viewDidLoad() {
16         super.viewDidLoad()
17
18         self.setupHamburgerUI()
19         // Do any additional setup after loading the view.
20     }
21
22     private func setupHamburgerUI()
23     {
24         self.mainBackgroundView.layer.cornerRadius = 40
25         self.mainBackgroundView.clipsToBounds = true
26
27         self.profilePictureImage.layer.cornerRadius = 40
28         self.profilePictureImage.clipsToBounds = true
29     }
30
31     @IBAction func clickedOnButton(_ sender: Any) {
32         self.delegate?.hideHamburgerMenu()
33     }
34
35     @IBAction func Contactanos(_ sender: UIButton) {
36         let alerta = UIAlertController(title: "Contactanos!", message: "alu.18131215@correo.itlalaguna.edu.mx", preferredStyle: .alert)
37         let accionCancelar = UIAlertAction(title: "Cancelar", style: .destructive)
38         let accionAceptar = UIAlertAction(title: "Aceptar", style: .default) { _ in
39             print("Correo")
40         }
41
42         alerta.addAction(accionCancelar)
43         alerta.addAction(accionAceptar)
44
45         present(alerta, animated: true)
46         self.delegate?.hideHamburgerMenu()
47     }
48
49     @IBAction func Acercadebutton(_ sender: UIButton) {
50
51         let alerta = UIAlertController(title: "CEAG!", message: "Empresa dedicada a mejorar estilos de vida", preferredStyle: .alert)
52         let accionCancelar = UIAlertAction(title: "Cancelar", style: .destructive)
53         let accionAceptar = UIAlertAction(title: "Aceptar", style: .default) { _ in
54             print("Correo")
55         }
56
57         alerta.addAction(accionCancelar)
58         alerta.addAction(accionAceptar)
59
60         present(alerta, animated: true)
```

## View controller

```
3 // Health-DeliverySW
4 //
5 // C
6 //
7
8 import UIKit
9 import Firebase
10 import GoogleSignIn
11 class ViewController: UIViewController {
12
13     @IBOutlet weak var CorreoTextField: UITextField!
14
15     @IBOutlet weak var Google: UIButton!
16     @IBOutlet weak var ContraseñaTextField: UITextField!
17
18     override func viewDidLoad() {
19         super.viewDidLoad()
20         // Do any additional setup after loading the view.
21     }
22
23     @IBAction func LoginButton(_ sender: UIButton) {
24         guard let correo = CorreoTextField.text else {return}
25         guard let contra = ContraseñaTextField.text else {return}
26
27         Auth.auth().signIn(withEmail: correo, password: contra) {firebaseResult, error in
28             if let e = error {
29                 print("Error")
30             }
31             else {
32                 //Regresar a la primera interfaz
33                 self.performSegue(withIdentifier: "Siguiente", sender: self)
34             }
35         }
36     }
37 }
38
39
40 /* @IBAction func SignGoogle(_ sender: UIButton) {
41
42     guard let clientID = FirebaseApp.app()?.options.clientID else { return }
43
44     // Create Google Sign In configuration object.
45     let config = GIDConfiguration(clientID: clientID)
46     GIDSignIn.sharedInstance.configuration = config
47
48     // Start the sign in flow!
49     GIDSignIn.sharedInstance.signIn(withPresenting: self) { [unowned self] result, error in
50         guard error == nil else {
51
52         }
53
54         guard let user = result?.user,
55               let idToken = user.idToken?.tokenString
56         else {
57             // ...
58         }
59
60         let credential = GoogleAuthProvider.credential(withIDToken: idToken,
```

## Creacion cuenta view controller

```
1 //
2 // CreacionCuentaViewController.swift
3 // Health-DeliverySW
4 //
5 // Created by Daniel Saldivar on 09/05/23.
6 //
7
8 import UIKit
9 import Firebase
10
11 class CreacionCuentaViewController: UIViewController {
12
13     @IBOutlet weak var CCorreoTextField: UITextField!
14     @IBOutlet weak var CContraseñaTextField: UITextField!
15
16     override func viewDidLoad() {
17         super.viewDidLoad()
18     }
19
20     @IBAction func CrearCuenta(_ sender: UIButton) {
21         guard let correo = CCorreoTextField.text else {return}
22         guard let contra = CContraseñaTextField.text else {return}
23
24         Auth.auth().createUser(withEmail: correo, password: contra) {firebaseResult, error in
25             if let e = error {
26                 print("error")
27             }
28             else{
29                 self.performSegue(withIdentifier: "Siguiente", sender: self)
30             }
31         }
32     }
33 }
34
35
36
37 }
38
```



## Slide menú view controller

```
1 //
2 // SlideMenu.swift
3 // Health-DeliverySW
4 //
5 // Created by Daniel Saldivar on 20/05/23.
6 //
7
8 import UIKit
9
10 class SlideMenu: UIViewController, UITableViewDelegate, UITableViewDataSource, HamburgerViewControllerDelegate {
11     @IBOutlet weak var mainBackView: UIView!
12     @IBOutlet weak var hamburgerView: UIView!
13     @IBOutlet weak var leadingConstraintForHamburgerView: NSLayoutConstraint!
14
15     @IBOutlet weak var backViewForHamburger: UIView!
16
17     override func viewDidLoad() {
18         super.viewDidLoad()
19         // Do any additional setup after loading the view.
20         self.backViewForHamburger.isHidden = true
21         self.mainBackView.layer.cornerRadius = 40
22         self.mainBackView.clipsToBounds = true
23     }
24
25     @IBAction func tappedOnHamburgerbackView(_ sender: Any) {
26         self.hideHamburgerView()
27     }
28
29     func hideHamburgerMenu() {
30         self.hideHamburgerView()
31     }
32
33     private func hideHamburgerView()
34     {
35         UIView.animate(withDuration: 0.1) {
36             self.leadingConstraintForHamburgerView.constant = 10
37             self.view.layoutIfNeeded()
38         } completion: { (status) in
39             self.backViewForHamburger.alpha = 0.0
40             UIView.animate(withDuration: 0.1) {
41                 self.leadingConstraintForHamburgerView.constant = -280
42                 self.view.layoutIfNeeded()
43             } completion: { (status) in
44                 self.backViewForHamburger.isHidden = true
45                 self.isHamburgerMenuShown = false
46             }
47         }
48     }
49
50     @IBAction func showHamburgerMenu(_ sender: Any) {
51         UIView.animate(withDuration: 0.1) {
52             self.leadingConstraintForHamburgerView.constant = 10
53             self.view.layoutIfNeeded()
54         } completion: { (status) in
55             self.backViewForHamburger.alpha = 0.75
56             self.backViewForHamburger.isHidden = false
57             UIView.animate(withDuration: 0.1) {
58                 self.leadingConstraintForHamburgerView.constant = 0
59             }
60         }
61     }
62
63     private func hideHamburgerView()
64     {
65         self.view.layoutIfNeeded()
66         } completion: { (status) in
67             self.backViewForHamburger.isHidden = true
68             self.isHamburgerMenuShown = false
69         }
70     }
71
72     @IBAction func showHamburgerMenu(_ sender: Any) {
73         UIView.animate(withDuration: 0.1) {
74             self.leadingConstraintForHamburgerView.constant = 10
75             self.view.layoutIfNeeded()
76         } completion: { (status) in
77             self.backViewForHamburger.alpha = 0.75
78             self.backViewForHamburger.isHidden = false
79             UIView.animate(withDuration: 0.1) {
80                 self.leadingConstraintForHamburgerView.constant = 0
81                 self.view.layoutIfNeeded()
82             } completion: { (status) in
83                 self.isHamburgerMenuShown = true
84             }
85         }
86         self.backViewForHamburger.isHidden = false
87     }
88
89     var hamburgerViewController: HamburgerViewController?
90
91     override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
92         if (segue.identifier == "hamburgerSegue") {
93             if let controller = segue.destination as? HamburgerViewController {
94                 self.hamburgerViewController = controller
95                 self.hamburgerViewController?.delegate = self
96             }
97         }
98     }
99
100     func tableView(_ tableView: UITableView, numberOfRowsInSectionSection section: Int) -> Int {
101         return 30
102     }
103
104     func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
105         let cell: MovieTableViewCell = tableView.dequeueReusableCell(withIdentifier: "MovieTableViewCell", for: indexPath) as!
106         MovieTableViewCell
107         let cell1: PastaTableViewCell = tableView.dequeueReusableCell(withIdentifier: "PastaTableViewCell", for: indexPath) as! PastaTableViewCell
108         let cell2: EnsaladaTableViewCell = tableView.dequeueReusableCell(withIdentifier: "EnsaladaTableViewCell", for: indexPath) as!
109         EnsaladaTableViewCell
110         let cell3: AgriTableViewCell = tableView.dequeueReusableCell(withIdentifier: "AgriTableViewCell", for: indexPath) as! AgriTableViewCell
111
112         cell.selectionStyle = .none
113         cell.backView.layer.cornerRadius = 8
114     }
```

```

10 class SlideMenu: UIViewController, UITableViewDelegate, UITableViewDataSource, HamburgerViewControllerDelegate {
87 func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
88     cell.profilePicImage.layer.cornerRadius = 20
99     cell.profilePicImage.clipsToBounds = true
100
101     return cell
102     return cell1
103     return cell2
104     return cell3
105 }
106
107 private var isHamburgerMenuShown: Bool = false
108 private var beginPoint: CGFloat = 0.0
109 private var difference: CGFloat = 0.0
110
111 override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {
112     if (isHamburgerMenuShown)
113     {
114         if let touch = touches.first
115         {
116             let location = touch.location(in: backViewForHamburger)
117             beginPoint = location.x
118         }
119     }
120 }
121
122 override func touchesMoved(_ touches: Set<UITouch>, with event: UIEvent?) {
123     if (isHamburgerMenuShown)
124     {
125         if let touch = touches.first
126         {
127             let location = touch.location(in: backViewForHamburger)
128
129             let differenceFromBeginPoint = beginPoint - location.x
130
131             if (differenceFromBeginPoint > 0 || differenceFromBeginPoint < 280)
132             {
133                 difference = differenceFromBeginPoint
134                 self.leadingConstraintForHamburgerView.constant = -differenceFromBeginPoint
135                 self.backViewForHamburger.alpha = 0.75 - (0.75 * differenceFromBeginPoint / 280)
136             }
137         }
138     }
139 }
140
141 override func touchesEnded(_ touches: Set<UITouch>, with event: UIEvent?) {
142     if (isHamburgerMenuShown)
143     {
144         if (difference > 140)
145         {
146             UIView.animate(withDuration: 0.1) {
147                 self.leadingConstraintForHamburgerView.constant = -290
148             } completion: { (status) in
149                 self.backViewForHamburger.alpha = 0.0
150                 self.isHamburgerMenuShown = false
151                 self.backViewForHamburger.isHidden = true
152             }
153         }
154         else{

```

⚠ Code after 'return' will never be executed

## CalenMensualController

```
8 import UIKit
9
10 class CalenMensualController: UIViewController, UICollectionViewDelegate, UICollectionViewDataSource
11 {
12     @IBOutlet weak var monthLabel: UILabel!
13     @IBOutlet weak var collectionView: UICollectionView!
14
15     var totalSquares = [CalendarDay]()
16
17     override func viewDidLoad()
18     {
19         super.viewDidLoad()
20         setCellsView()
21         setMonthView()
22     }
23
24     func setCellsView()
25     {
26         let width = (collectionView.frame.size.width - 2) / 8
27         let height = (collectionView.frame.size.height - 2) / 8
28
29         let flowLayout = collectionView.collectionViewLayout as! UICollectionViewFlowLayout
30         flowLayout.itemSize = CGSize(width: width, height: height)
31     }
32
33     func setMonthView()
34     {
35         totalSquares.removeAll()
36
37         let daysInMonth = CalendarHelper().daysInMonth(date: selectedDate)
38         let firstDayOfMonth = CalendarHelper().firstOfMonth(date: selectedDate)
39         let startingSpaces = CalendarHelper().weekDay(date: firstDayOfMonth)
40
41         let prevMonth = CalendarHelper().minusMonth(date: selectedDate)
42         let daysInPrevMonth = CalendarHelper().daysInMonth(date: prevMonth)
43
44         var count: Int = 1
45
46         while(count <= 42)
47         {
48             let calendarDay = CalendarDay()
49             if count <= startingSpaces
50             {
51                 let prevMonthDay = daysInPrevMonth - startingSpaces + count
52                 calendarDay.day = String(prevMonthDay)
53                 calendarDay.month = CalendarDay.Month.previous
54             }
55             else if count - startingSpaces > daysInMonth
56             {
57                 calendarDay.day = String(count - daysInMonth - startingSpaces)
58
59                 calendarDay.day = String(count - daysInMonth - startingSpaces)
60                 calendarDay.month = CalendarDay.Month.next
61             }
62             else
63             {
64                 calendarDay.day = String(count - startingSpaces)
65                 calendarDay.month = CalendarDay.Month.current
66             }
67             totalSquares.append(calendarDay)
68             count += 1
69         }
70
71         monthLabel.text = CalendarHelper().monthString(date: selectedDate)
72         + " " + CalendarHelper().yearString(date: selectedDate)
73         collectionView.reloadData()
74     }
75
76     func collectionView(_ collectionView: UICollectionView, numberOfItemsInSection section: Int) -> Int {
77         totalSquares.count
78     }
79
80     func collectionView(_ collectionView: UICollectionView, cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {
81         let cell = collectionView.dequeueReusableCell(withReuseIdentifier: "calCell", for: indexPath) as! CalendarCell
82
83         let calendarDay = totalSquares[indexPath.item]
84
85         cell.dayOfMonth.text = calendarDay.day
86         if(calendarDay.month == CalendarDay.Month.current)
87         {
88             cell.dayOfMonth.textColor = UIColor.black
89         }
90         else
91         {
92             cell.dayOfMonth.textColor = UIColor.gray
93         }
94
95         return cell
96     }
97
98     @IBAction func previousMonth(_ sender: Any)
99     {
100         selectedDate = CalendarHelper().minusMonth(date: selectedDate)
101         setMonthView()
102     }
103
104     @IBAction func nextMonth(_ sender: Any)
105     {
106         selectedDate = CalendarHelper().plusMonth(date: selectedDate)
107         setMonthView()
108     }
109
110     override open var shouldAutorotate: Bool
111     {
112         return false
113     }
114 }
```



## Calen Semanal

```
1 import UIKit
2
3 var selectedDate = Date()
4
5 class WeeklyViewController: UIViewController, UICollectionViewDelegate, UICollectionViewDataSource,
6     UITableViewDelegate, UITableViewDataSource
7 {
8
9     @IBOutlet weak var monthLabel: UILabel!
10    @IBOutlet weak var tableView: UITableView!
11    @IBOutlet weak var collectionView: UICollectionView!
12
13
14    var totalSquares = [Date]()
15
16
17    override func viewDidLoad()
18    {
19        super.viewDidLoad()
20        setCellsView()
21        setWeekView()
22    }
23
24    func setCellsView()
25    {
26        let width = (collectionView.frame.size.width - 2) / 8
27        let height = (collectionView.frame.size.height - 2)
28
29        let flowLayout = collectionView.collectionViewLayout as! UICollectionViewFlowLayout
30        flowLayout.itemSize = CGSize(width: width, height: height)
31    }
32
33    func setWeekView()
34    {
35        totalSquares.removeAll()
36
37        var current = CalendarHelper().sundayForDate(date: selectedDate)
38        let nextSunday = CalendarHelper().addDays(date: current, days: 7)
39
40        while (current < nextSunday)
41        {
42            totalSquares.append(current)
43            current = CalendarHelper().addDays(date: current, days: 1)
44        }
45
46        monthLabel.text = CalendarHelper().monthString(date: selectedDate)
47        + " " + CalendarHelper().yearString(date: selectedDate)
48        collectionView.reloadData()
49        tableView.reloadData()
50    }
51
52    func collectionView(_ collectionView: UICollectionView, numberOfItemsInSection section: Int) -> Int {
53        totalSquares.count
54    }
55
56    func collectionView(_ collectionView: UICollectionView, cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {
57        let cell = collectionView.dequeueReusableCell(withReuseIdentifier: "calCell", for: indexPath) as! CalendarCell
58
59        let date = totalSquares[indexPath.item]
60
61        class WeeklyViewController: UIViewController, UICollectionViewDelegate, UICollectionViewDataSource,
62            func collectionView(_ collectionView: UICollectionView, cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {
63            let cell = collectionView.dequeueReusableCell(withReuseIdentifier: "calCell", for: indexPath) as! CalendarCell
64
65            let date = totalSquares[indexPath.item]
66            cell.dayOfMonth.text = String(CalendarHelper().dayOfMonth(date: date))
67
68            if(date == selectedDate)
69            {
70                cell.backgroundColor = UIColor.systemGreen
71            }
72            else
73            {
74                cell.backgroundColor = UIColor.white
75            }
76
77            return cell
78        }
79
80        func collectionView(_ collectionView: UICollectionView, didSelectItemAt indexPath: IndexPath)
81        {
82            selectedDate = totalSquares[indexPath.item]
83            collectionView.reloadData()
84            tableView.reloadData()
85        }
86
87        @IBAction func previousWeek(_ sender: Any)
88        {
89            selectedDate = CalendarHelper().addDays(date: selectedDate, days: -7)
90            setWeekView()
91        }
92
93        @IBAction func nextWeek(_ sender: Any)
94        {
95            selectedDate = CalendarHelper().addDays(date: selectedDate, days: 7)
96            setWeekView()
97        }
98
99        override open var shouldAutorotate: Bool
100        {
101            return false
102        }
103
104        func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int
105        {
106            return Event().eventsForDate(date: selectedDate).count
107        }
108
109        func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell
110        {
111            let cell = tableView.dequeueReusableCell(withIdentifier: "cellID") as! EventCell
112            let event = Event().eventsForDate(date: selectedDate)[indexPath.row]
113            cell.eventLabel.text = event.name + " " + CalendarHelper().timeString(date: event.date)
114            return cell
115        }
116    }
```



## DailyEvent

```
1 import UIKit
2
3
4 class DailyViewController: UIViewController, UITableViewDelegate, UITableViewDataSource
5 {
6     @IBOutlet weak var hourTableView: UITableView!
7     @IBOutlet weak var dayOfWeekLabel: UILabel!
8     @IBOutlet weak var dayLabel: UILabel!
9
10    var hours = [Int]()
11
12    override func viewDidLoad()
13    {
14        super.viewDidLoad()
15        initTime()
16        setDayView()
17    }
18
19    func initTime()
20    {
21        for hour in 0...23
22        {
23            hours.append(hour)
24        }
25    }
26
27    func setDayView()
28    {
29        dayLabel.text = CalendarHelper().monthDayString(date: selectedDate)
30        dayOfWeekLabel.text = CalendarHelper().weekDayAsString(date: selectedDate)
31        hourTableView.reloadData()
32    }
33
34    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
35        return hours.count
36    }
37
38    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
39        let cell = tableView.dequeueReusableCell(withIdentifier: "cellDailyID") as! DailyCell
40
41        let hour = hours[indexPath.row]
42        cell.time.text = formatHour(hour: hour)
43
44        let events = Event().eventsForDateAndTime(date: selectedDate, hour: hour)
45        setEvents(cell, events)
46
47        return cell
48    }
49
50    func setEvents(_ cell: DailyCell, _ events: [Event])
51    {
52        hideAll(cell)
53        switch events.count
54        {
55            case 1:
56                setEvent1(cell, events[0])
57            case 2:
58                setEvent1(cell, events[0])
59                setEvent2(cell, events[1])
60        }
61
62        class DailyViewController: UIViewController, UITableViewDelegate, UITableViewDataSource
63        func setEvents(_ cell: DailyCell, _ events: [Event])
64        {
65            case 3:
66                setEvent1(cell, events[0])
67                setEvent2(cell, events[1])
68                setEvent3(cell, events[2])
69
70            case let count where count > 3:
71                setEvent1(cell, events[0])
72                setEvent2(cell, events[1])
73                setMoreEvents(cell, events.count - 2)
74            default:
75                break
76        }
77    }
78
79    func setMoreEvents(_ cell: DailyCell, _ count: Int)
80    {
81        cell.event3.isHidden = false
82        cell.event3.text = String(count) + " More Events"
83    }
84
85    func setEvent1(_ cell: DailyCell, _ event: Event)
86    {
87        cell.event1.isHidden = false
88        cell.event1.text = event.name
89    }
90
91    func setEvent2(_ cell: DailyCell, _ event: Event)
92    {
93        cell.event2.isHidden = false
94        cell.event2.text = event.name
95    }
96
97    func setEvent3(_ cell: DailyCell, _ event: Event)
98    {
99        cell.event3.isHidden = false
100        cell.event3.text = event.name
101    }
102
103    func hideAll(_ cell: DailyCell)
104    {
105        cell.event1.isHidden = true
106        cell.event2.isHidden = true
107        cell.event3.isHidden = true
108    }
109
110    func formatHour(hour: Int) -> String
111    {
112        return String(format: "%02d:%02d", hour, 0)
113    }
114
115    @IBAction func nextDayAction(_ sender: Any)
116    {
117        selectedDate = CalendarHelper().addDays(date: selectedDate, days: 1)
118        setDayView()
119    }
120 }
```

## CalendarHelper

```
1 import Foundation
2 import UIKit
3
4 class CalendarHelper
5 {
6     let calendar = Calendar.current
7
8     func plusMonth(date: Date) -> Date
9     {
10         return calendar.date(byAdding: .month, value: 1, to: date)!
11     }
12
13     func minusMonth(date: Date) -> Date
14     {
15         return calendar.date(byAdding: .month, value: -1, to: date)!
16     }
17
18     func monthString(date: Date) -> String
19     {
20         let dateFormatter = DateFormatter()
21         dateFormatter.dateFormat = "LLLL"
22         return dateFormatter.string(from: date)
23     }
24
25     func monthDayString(date: Date) -> String
26     {
27         let dateFormatter = DateFormatter()
28         dateFormatter.dateFormat = "LLLL dd"
29         return dateFormatter.string(from: date)
30     }
31
32     func yearString(date: Date) -> String
33     {
34         let dateFormatter = DateFormatter()
35         dateFormatter.dateFormat = "yyyy"
36         return dateFormatter.string(from: date)
37     }
38
39     func timeString(date: Date) -> String
40     {
41         let dateFormatter = DateFormatter()
42         dateFormatter.dateFormat = "HH:mm"
43         return dateFormatter.string(from: date)
44     }
45
46     func daysInMonth(date: Date) -> Int
47     {
48         let range = calendar.range(of: .day, in: .month, for: date)!
49         return range.count
50     }
51
52     func dayOfMonth(date: Date) -> Int
53     {
54         let components = calendar.dateComponents([.day], from: date)
55         return components.day!
56     }
57
58     func hourFromDate(date: Date) -> Int
59     {
60         let components = calendar.dateComponents([.hour], from: date)
61         return components.hour!
62     }
63
64     func firstOfMonth(date: Date) -> Date
65     {
66         let components = calendar.dateComponents([.year, .month], from: date)
67         return calendar.date(from: components)!
68     }
69
70     func weekDay(date: Date) -> Int
71     {
72         let components = calendar.dateComponents([.weekday], from: date)
73         return components.weekday! - 1
74     }
75
76     func weekDayAsString(date: Date) -> String
77     {
78         switch weekDay(date: date)
79         {
80             case 0:
81                 return "Sunday"
82             case 1:
83                 return "Monday"
84             case 2:
85                 return "Tuesday"
86             case 3:
87                 return "Wednesday"
88             case 4:
89                 return "Thursday"
90             case 5:
91                 return "Friday"
92             case 6:
93                 return "Saturday"
94             default:
95                 return ""
96         }
97     }
98
99     func addDays(date: Date, days: Int) -> Date
100     {
101         return calendar.date(byAdding: .day, value: days, to: date)!
102     }
103
104     func sundayForDate(date: Date) -> Date
105     {
106         var current = date
107         let oneWeekAgo = addDays(date: current, days: -7)
108         while(current > oneWeekAgo)
109         {
110             current = addDays(date: current, days: -1)
111             if current.weekDay() == 0 {
112                 return current
113             }
114         }
115     }
116 }
```

## EventEdit

```
1 import UIKit
2 //Metodo que funciona para poder agregar y modificar eventos en calendario
3 class EventEditViewController: UIViewController
4 {
5
6     @IBOutlet weak var nameTF: UITextField!
7     @IBOutlet weak var datePicker: UIDatePicker!
8
9     override func viewDidLoad()
10     {
11         super.viewDidLoad()
12
13         datePicker.date = selectedDate
14     }
15
16     @IBAction func saveAction(_ sender: Any)
17     {
18         let newEvent = Event()
19         newEvent.id = eventsList.count
20         newEvent.name = nameTF.text
21         newEvent.date = datePicker.date
22         eventsList.append(newEvent)
23         navigationController?.popViewController(animated: true)
24     }
25 }
26
```

## Calendar Day

```
1 import Foundation
2 // Metodo que funciona para asignar tipo de dato a los meses, semana y mes
3 class CalendarDay
4 {
5     var day: String!
6     var month: Month!
7
8     enum Month
9     {
10         case previous
11         case current
12         case next
13     }
14 }
15
```

## Event

```
1
2 import Foundation
3
4 var eventsList = [Event]()
5 //Metodo para creacion de evento desde boton agregar perfil
6 class Event
7 {
8     var id: Int!
9     var name: String!
10    var date: Date!
11
12    func eventsForDate(date: Date) -> [Event]
13    {
14        var daysEvents = [Event]()
15        for event in eventsList
16        {
17            if(Calendar.current.isDate(event.date, inSameDayAs:date))
18            {
19                daysEvents.append(event)
20            }
21        }
22        return daysEvents
23    }
24    //Metodo que funciona para asignar el día y hora al evento
25    func eventsForDateAndTime(date: Date, hour: Int) -> [Event]
26    {
27        var daysEvents = [Event]()
28        for event in eventsList
29        {
30            if(Calendar.current.isDate(event.date, inSameDayAs:date))
31            {
32                let eventHour = CalendarHelper().hourFromDate(date: event.date)
33                if eventHour == hour
34                {
35                    daysEvents.append(event)
36                }
37            }
38        }
39        return daysEvents
40    }
41 }
42
```