Advanced Software Engineering

Index

Name: Manish Parihar
Matriculation No - 874932

Project Name -->   Pocket Friend

1. Introduction -->

A project is based on completely machine learning (Artifical Intelligence) based , in which user can chat with autobot and reply according to their emotion. A question set will drill down to the app user based on their mood (either its good or bad).
Its just the prototype / gaming type which based on user emotion sets. Basically app will track your emotion / mood and according to that app machine learning optimiser will ask question more detailed question to you and based on your answer it will give suggestion to the user. This app is created to help and improve the mood of user.
Also app will show your result in about me section which indicated how many time your emotion were good or bad on date / month wise.
And which emotion set you chooses more based on the count and algorithm the bubble size will show bigger in about me.

2. User Requirement —>

To achieve user happy emotion provide a feature in app that will help to user to improve his emotion when he will be talking to his auto-chat. And track the user emotion set.
We need only tracking user name and date of birth from the user 212.

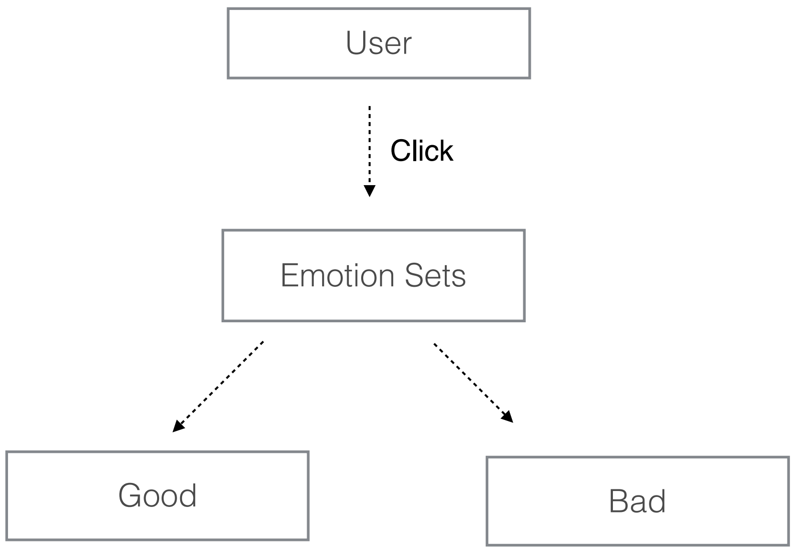
3.Technical Requirement -->

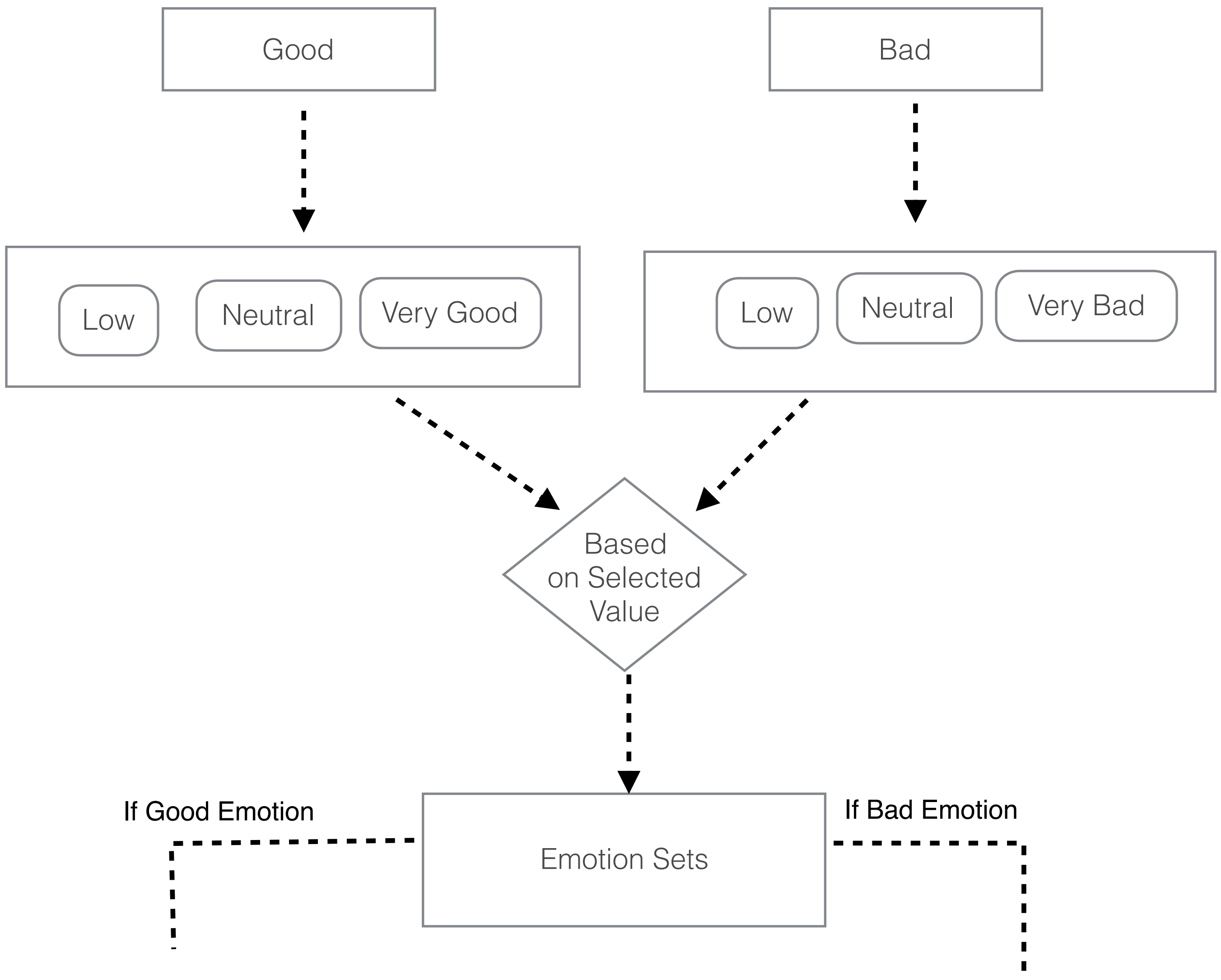
1. Xcode (iPhone/IOS app development)
2. Swift and Objective C Language
3. HTML Script for Mobile Web Page Launch
5. Sqlite 3 Database.

4. UML (Unified Modeling Language) —>

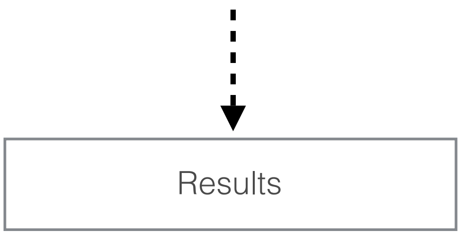
Set of notation elements that can be used to develop models for software systems. This concern the analysis, design and in general presentationn and documents.

Class Diagram ->

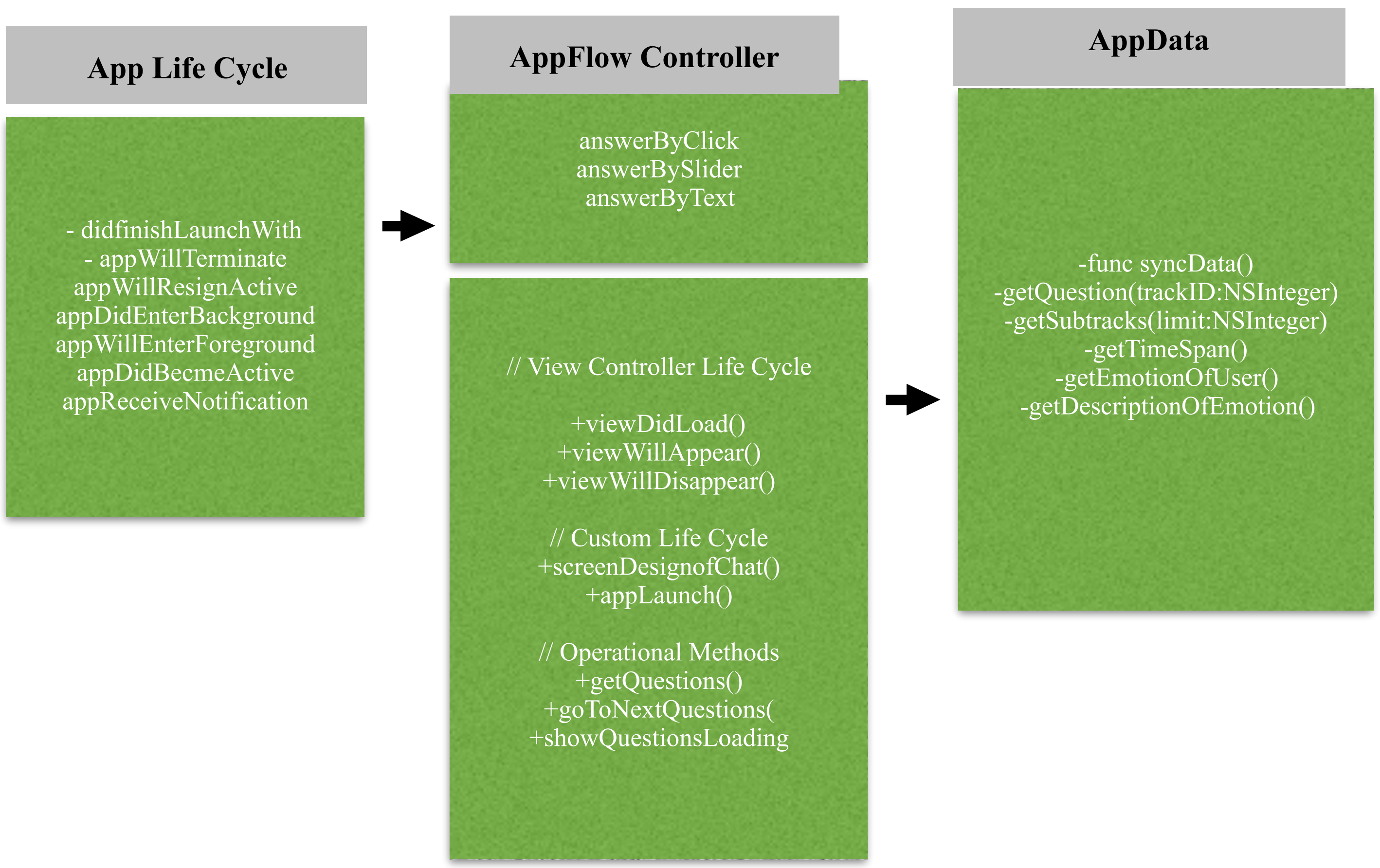




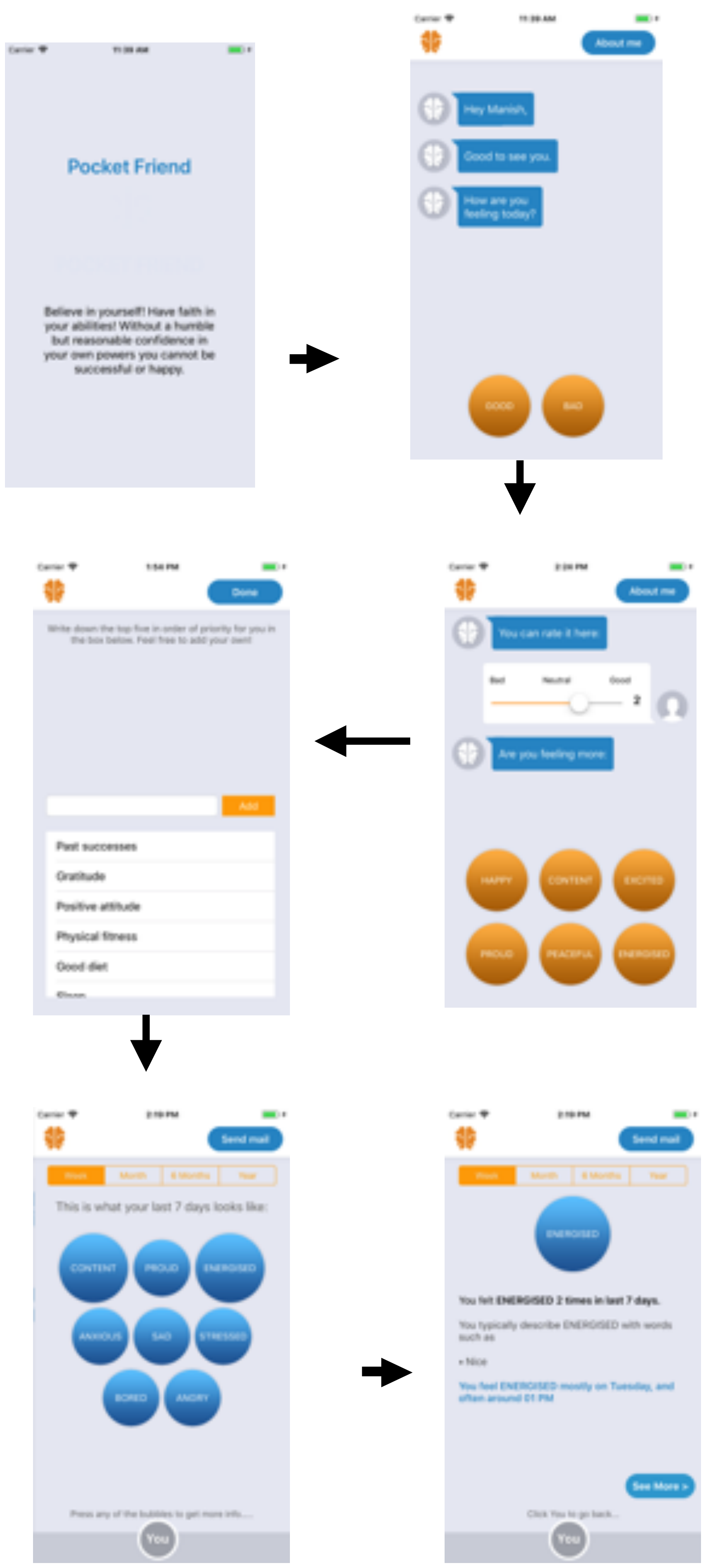
Good Emotion Sets	Bad Emotion Sets
Happy	Anxious
Content	Sad
Excited	Stressed
Proud	Bored
Peaceful	Flat
Energised	Angry



Static Structure Diagrams



App Diagram -
Displaying the app workflow



Believe in yourself! Have faith in your abilities! Without a humble but reasonable confidence in your own powers you cannot be successful or happy.

Hey Manish,

Good to see you.

How are you feeling today?

You can rate it here.

Are you feeling more:

Write down the top five in order of priority for you in the box below. Feel free to add your own!

Past successes
Gratitude
Positive attitude
Physical fitness
Good diet
Sleep

This is what your last 7 days looks like:

Press any of the bubbles to get more info....

You felt ENERGISED 2 times in last 7 days.

You typically describe ENERGISED with words such as

• Nice

You feel ENERGISED mostly on Tuesday, and often around 01 PM

Click You to go back...

5. Metrics -

Basically, there are many types of metrics related to software development. A software project consists of many elements,for example,

- Source code (Xcode counts line of codes)
- Binary Code
- External Library (Used 3 libraries)
    - (a) AlertView
    - (b) SMS Sending
    - (c) FMDB Database

Sonarcube is the probably best static code analyzer its easy to find bugs, code smell and security vulnerabilities.

As like sonarcube i used Fabric / Crashlytics - Its check the app performancece and health at every stage , issue or error tracker.

6. Clean Code Development :

(a) Coding Fundamental : Code Commenting and Versioning of Product

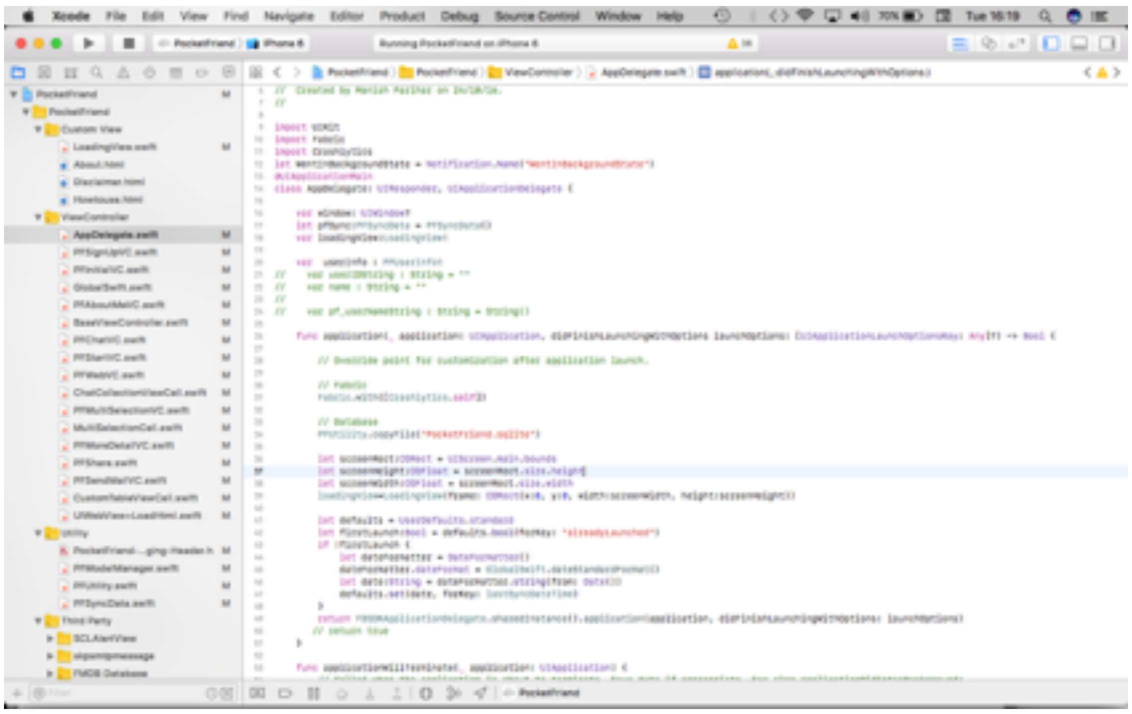
A proper naming conventions has to use insetad of bad code.
like an example :

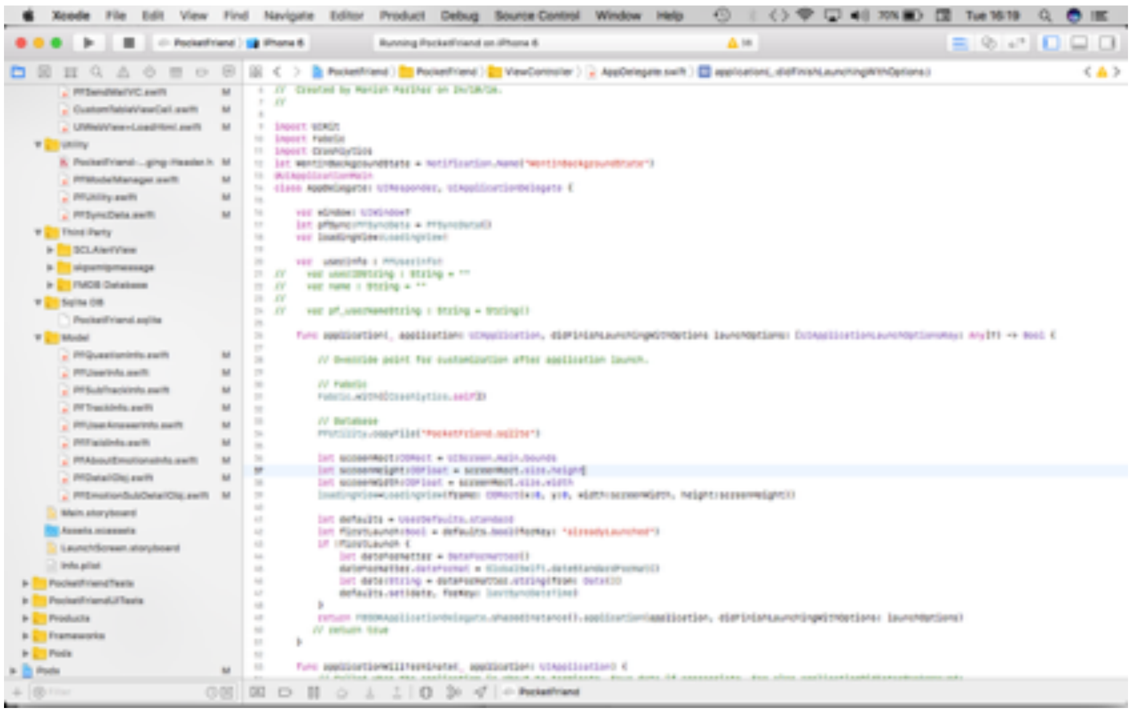
NSString *strUsername1 = “Code”; thats a bad syntax

Instead Use

NSString *string_Username = “Code”; well mentioned naming convention.

(b) Folder Structure : Always create folder according to the file name and if we have more than one file create separate folder for like web service, modals , view, controller etc please see the below image mentioned.

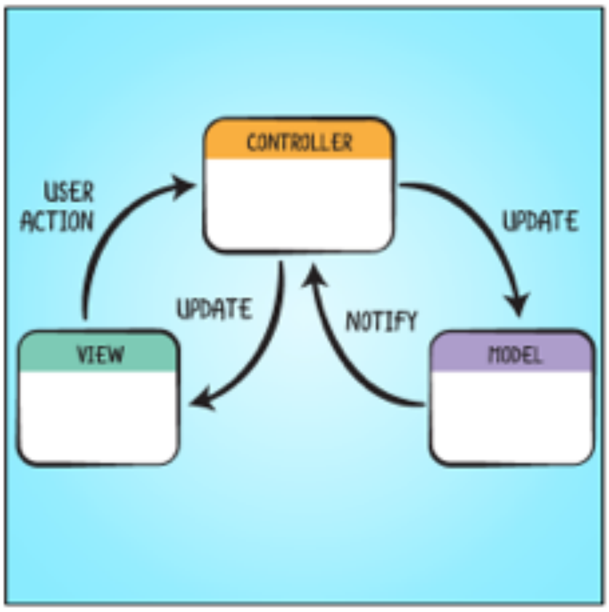




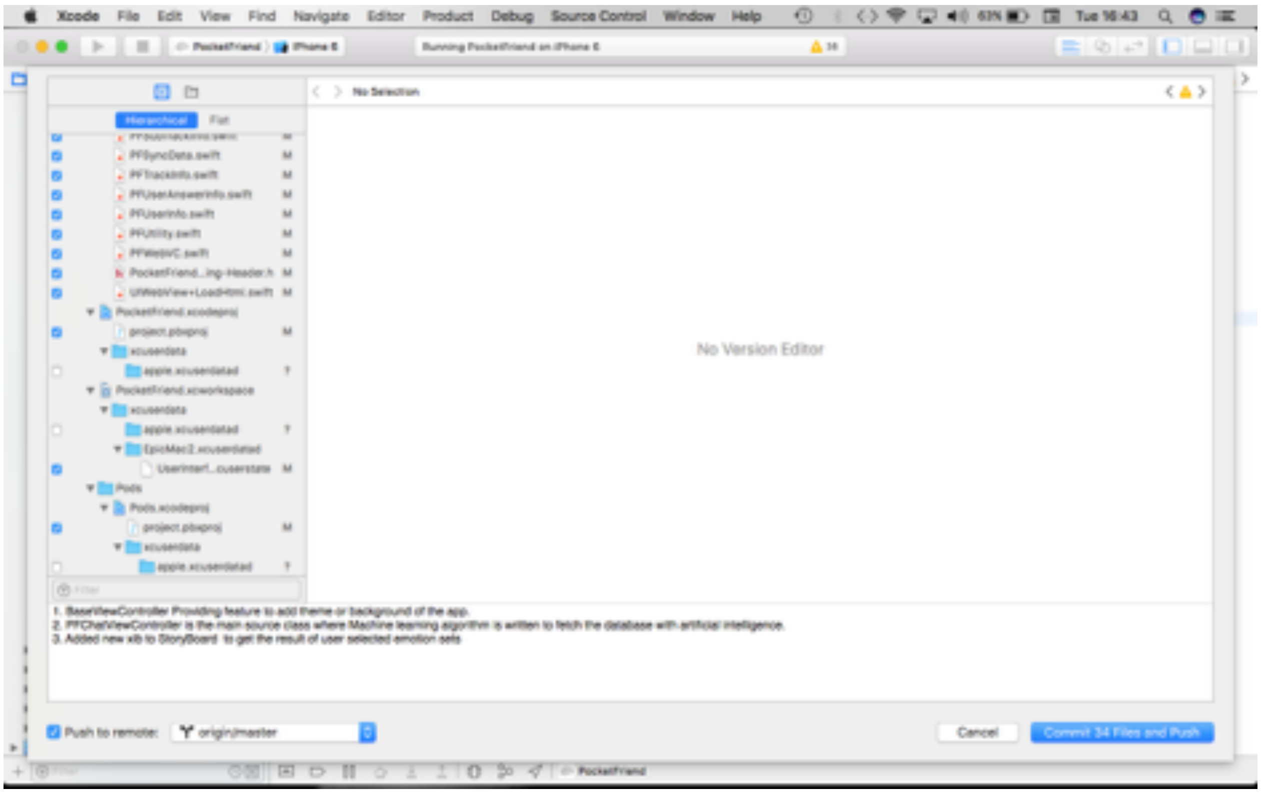
Model View Controller Architecture : Its made up of three layer, the model, the view and the controller

- Model is where your data resides, Things like persistence, model objects, parsers and networking code normally live there.

- View layer is the face of your app. Its classes are typically reusable, since there aren't domain specific login in them. Example - Label and TextView presents text on the screen, and its easily reusable.
- Controller mediates between the view and the model, typically via the delegation pattern.



6. **Continuous Delivery** - Jenkins is an open source automation server

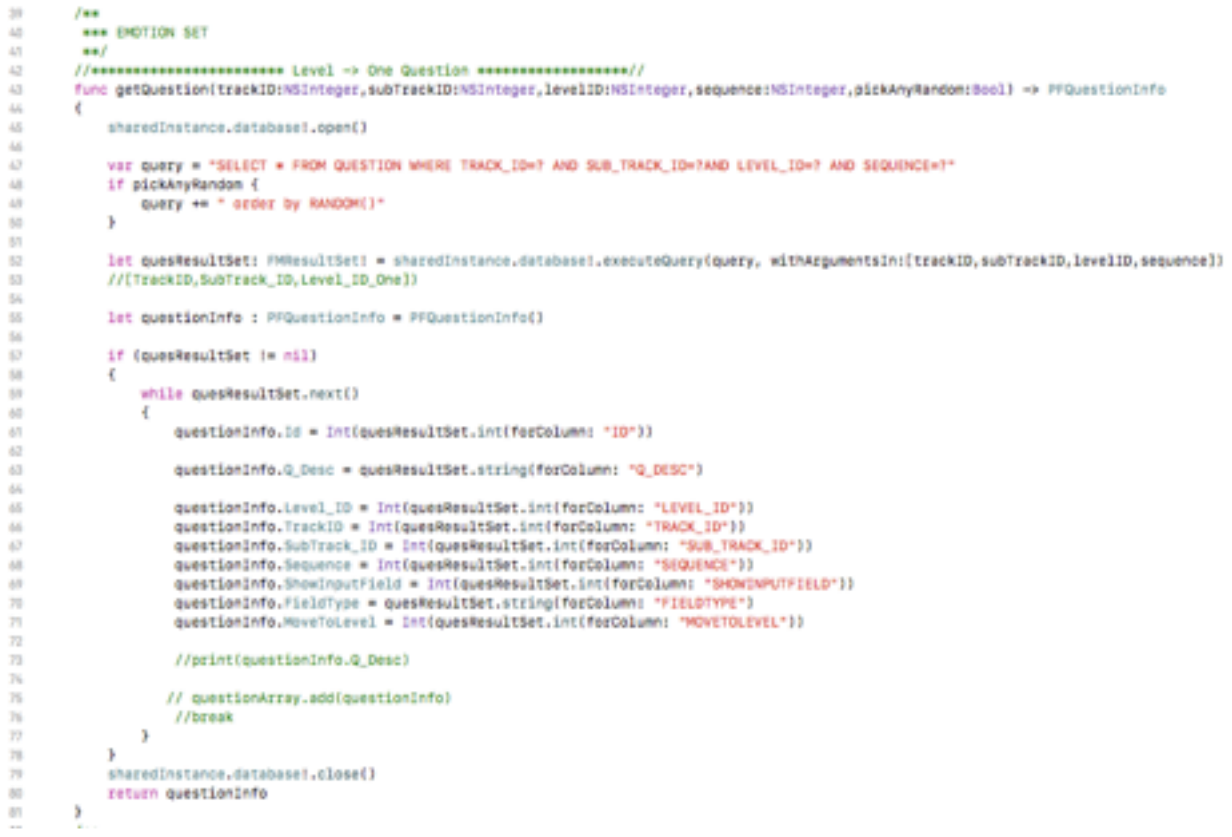


1. BaseViewController Providing feature to add theme or background of the app.
2. PFChatViewController is the main source class where Machine learning algorithm is written to fetch the database with artificial intelligence.
3. Added new xib to StoryBoard to get the result of user selected emotion sets

written in java. Its helps to automate the software development process and with the continuous integration and facilitating technical aspects of continuous delivery. Just like Git. Screen shot showing 34 files need to commit.

7. AOP

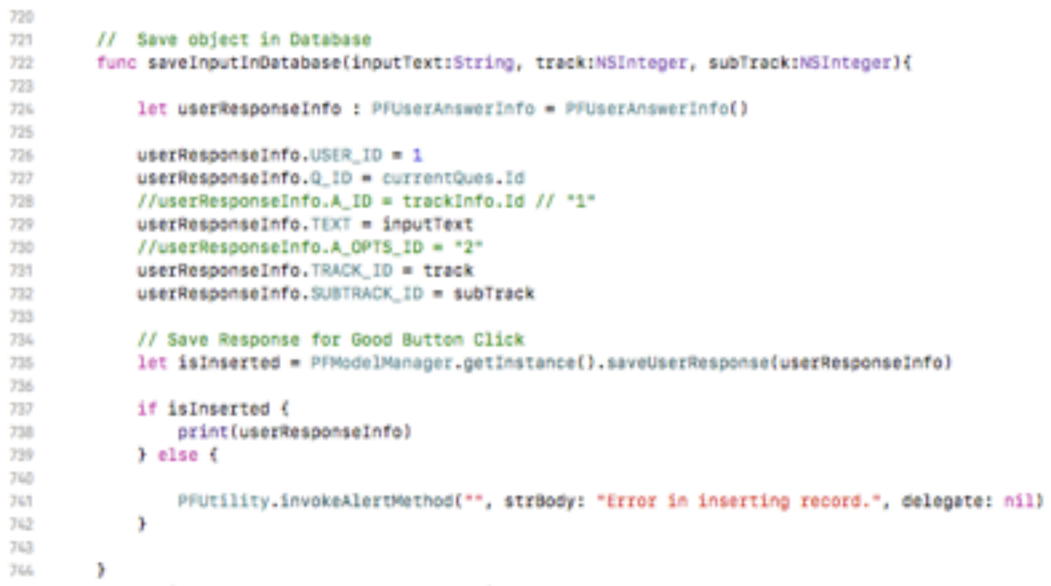
Aspect oriented is an approach to programng that allows global properties of the program to determinee how it is compiled into an executable program. AOP can be used with object-oriented programming (OOP). Its a common feature that’s typically scattered across methods, classes , object hierarchies or even entire object models. It should have structure.
For example , metrics is one common aspects



```
/**
 *** EMOTION SET
 **/
//************************ Level -> One Question ******************//
func getQuestion(trackID:NSInteger,subTrackID:NSInteger,levelID:NSInteger,sequence:NSInteger,pickAnyRandom:Bool) -> PFQuestionInfo
{
    sharedInstance.database!.open()

    var query = "SELECT * FROM QUESTION WHERE TRACK_ID=? AND SUB_TRACK_ID=?AND LEVEL_ID=? AND SEQUENCE=?"
    if pickAnyRandom {
        query += " order by RANDOM()"
    }

    let quesResultSet: FMResultSet! = sharedInstance.database!.executeQuery(query, withArgumentsIn:[trackID,subTrackID,levelID,sequence])
    //[TrackID,SubTrack_ID,Level_ID_One])

    let questionInfo : PFQuestionInfo = PFQuestionInfo()

    if (quesResultSet != nil)
    {
        while quesResultSet.next()
        {
            questionInfo.Id = Int(quesResultSet.int(forColumn: "ID"))

            questionInfo.Q_Desc = quesResultSet.string(forColumn: "Q_DESC")

            questionInfo.Level_ID = Int(quesResultSet.int(forColumn: "LEVEL_ID"))
            questionInfo.TrackID = Int(quesResultSet.int(forColumn: "TRACK_ID"))
            questionInfo.SubTrack_ID = Int(quesResultSet.int(forColumn: "SUB_TRACK_ID"))
            questionInfo.Sequence = Int(quesResultSet.int(forColumn: "SEQUENCE"))
            questionInfo.ShowInputField = Int(quesResultSet.int(forColumn: "SHOWINPUTFIELD"))
            questionInfo.FieldType = quesResultSet.string(forColumn: "FIELDTYPE")
            questionInfo.MoveToLevel = Int(quesResultSet.int(forColumn: "MOVETOLEVEL"))

            //print(questionInfo.Q_Desc)

           // questionArray.add(questionInfo)
            //break
        }
    }
    sharedInstance.database!.close()
    return questionInfo
}
```

8. DSL

As example : Class is PFUserAnswerInfo And Q_ID, USER_ID is object this class.

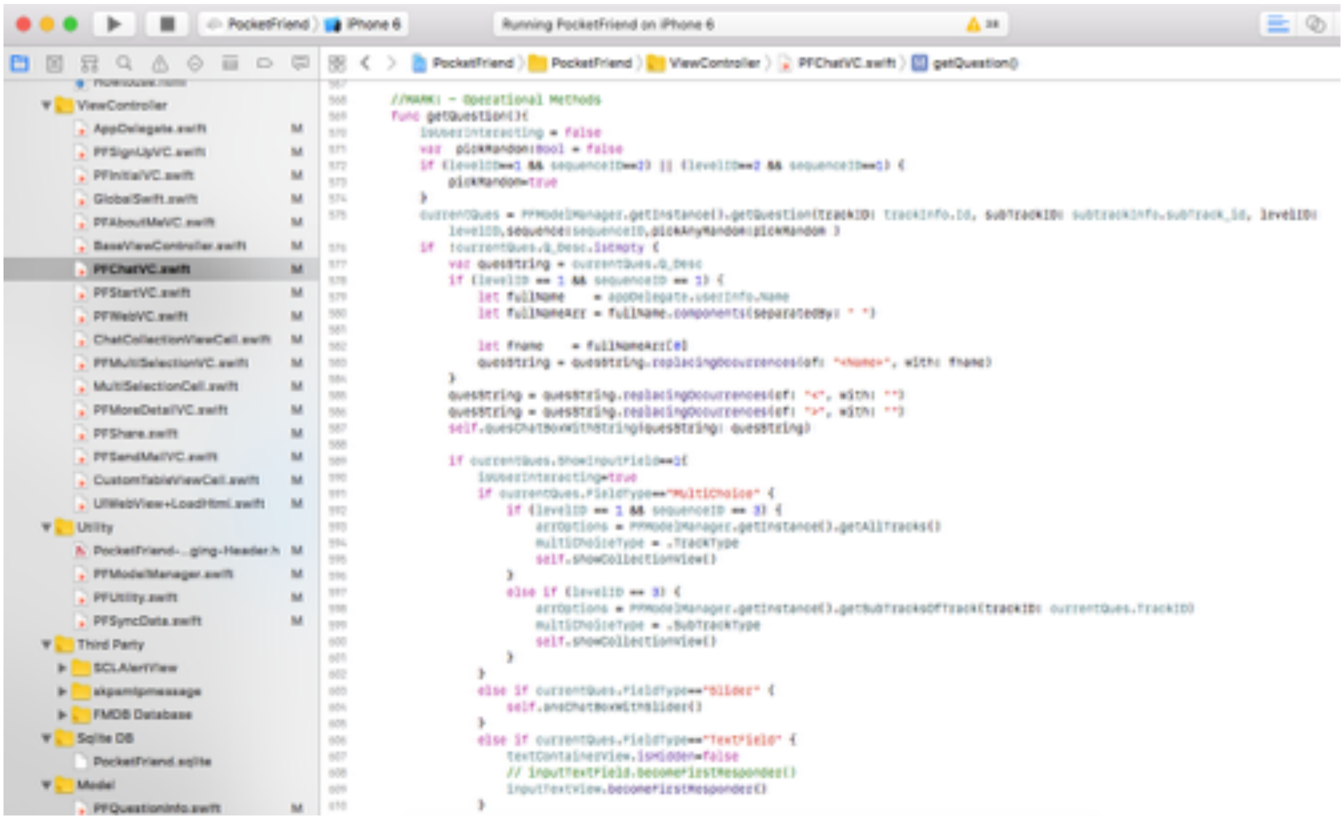


```
    //  Save object in Database
    func saveInputInDatabase(inputText:String, track:NSInteger, subTrack:NSInteger){

        let userResponseInfo : PFUserAnswerInfo = PFUserAnswerInfo()

        userResponseInfo.USER_ID = 1
        userResponseInfo.Q_ID = currentQues.Id
        //userResponseInfo.A_ID = trackInfo.Id // "1"
        userResponseInfo.TEXT = inputText
        //userResponseInfo.A_OPTS_ID = "2"
        userResponseInfo.TRACK_ID = track
        userResponseInfo.SUBTRACK_ID = subTrack

        // Save Response for Good Button Click
        let isInserted = PFModelManager.getInstance().saveUserResponse(userResponseInfo)

        if isInserted {
            print(userResponseInfo)
        } else {

            PFUtility.invokeAlertMethod("", strBody: "Error in inserting record.", delegate: nil)
        }

    }
```

9. Functional Programming —:

The main logic of my program is written on class PFChatVC.swift . It mentioned in Operational Methods in below image.

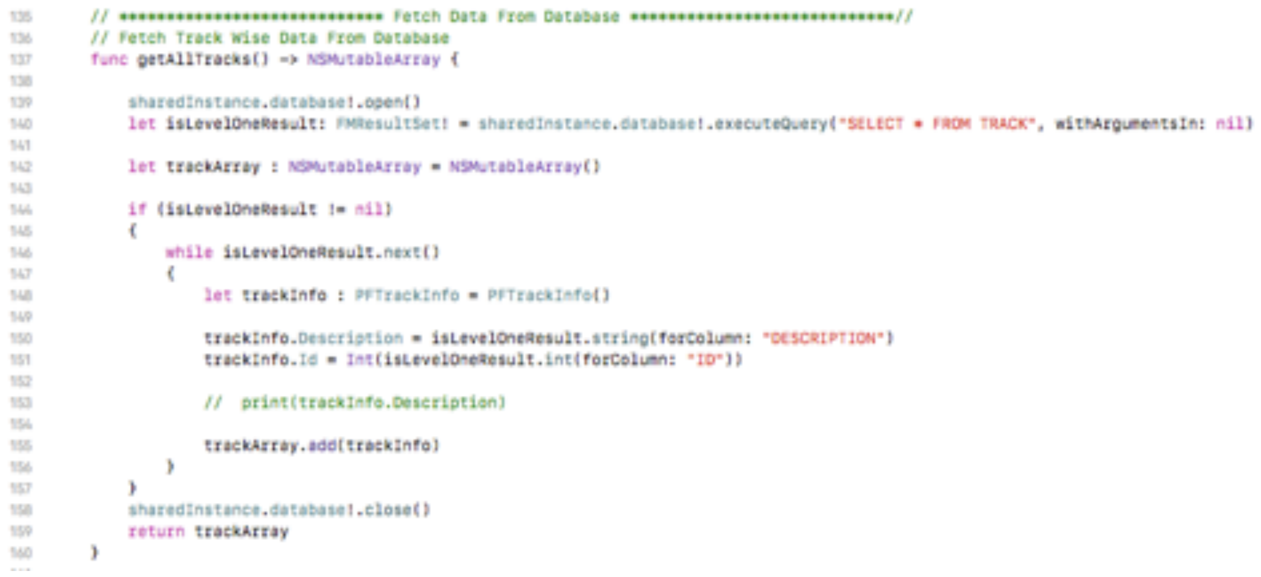
```
// MARK: - Operational Methods

 func getQuestion( )
{
       isUserInteracting = false
       var  pickRandom:Bool = false
       if (levelID==1 && sequenceID==2) || (levelID==2 && sequenceID==1) {
          pickRandom=true
       }
       currentQues = PFModelManager.getInstance().getQuestion(trackID: trackInfo.Id,
subTrackID: subtrackInfo.subTrack_id, levelID:
levelID,sequence:sequenceID,pickAnyRandom:pickRandom )
       if  !currentQues.Q_Desc.isEmpty {
          var quesString = currentQues.Q_Desc
          if (levelID == 1 && sequenceID == 1) {
              let fullName     = appDelegate.userInfo.Name
              let fullNameArr = fullName.components(separatedBy: " ")

              let fname     = fullNameArr[0]
              quesString = quesString.replacingOccurrences(of: "<Name>", with: fname)
          }
      }
```



```
//MARK: - Operational Methods
func getQuestion(){
    isUserInteracting = false
    var  pickRandom:Bool = false
    if (levelID==1 && sequenceID==2) || (levelID==2 && sequenceID==1) {
        pickRandom=true
    }
    currentQues = PFModelManager.getInstance().getQuestion(trackID: trackInfo.Id, subTrackID: subtrackInfo.subTrack_Id, levelID:
        levelID,sequence:sequenceID,pickAnyRandom:pickRandom )
    if  !currentQues.Q_Desc.isEmpty {
        var quesString = currentQues.Q_Desc
        if (levelID == 1 && sequenceID == 1) {
            let fullName     = appDelegate.userInfo.Name
            let fullNameArr = fullName.components(separatedBy: " ")

            let fname     = fullNameArr[0]
            quesString = quesString.replacingOccurrences(of: "<Name>", with: fname)
        }
        quesString = quesString.replacingOccurrences(of: "<", with: "")
        quesString = quesString.replacingOccurrences(of: ">", with: "")
        self.quesChatBoxWithString(quesString: quesString)

        if currentQues.ShowInputField==1{
            isUserInteracting=true
            if currentQues.FieldType=="MultiChoice" {
                if (levelID == 1 && sequenceID == 3) {
                    arrOptions = PFModelManager.getInstance().getAllTracks()
                    multiChoiceType = .TrackType
                    self.showCollectionView()
                }
                else if (levelID == 3) {
                    arrOptions = PFModelManager.getInstance().getSubTracksOfTrack(trackID: currentQues.TrackID)
                    multiChoiceType = .SubTrackType
                    self.showCollectionView()
                }
            }
            else if currentQues.FieldType=="Slider" {
                self.ansChatBoxWithSlider()
            }
            else if currentQues.FieldType=="TextField" {
                textContainerView.isHidden=false
                // inputTextField.becomeFirstResponder()
                inputTextView.becomeFirstResponder()
            }
```

10. Logical Approach

To fetch question set from the Pocketfriend database with sql query. And connect this query based function in another class function.



```
// **************************** Fetch Data From Database ****************************//
// Fetch Track Wise Data From Database
func getAllTracks() -> NSMutableArray {

    sharedInstance.database!.open()
    let isLevelOneResult: FMResultSet! = sharedInstance.database!.executeQuery("SELECT * FROM TRACK", withArgumentsIn: nil)

    let trackArray : NSMutableArray = NSMutableArray()

    if (isLevelOneResult != nil)
    {
        while isLevelOneResult.next()
        {
            let trackInfo : PFTrackInfo = PFTrackInfo()

            trackInfo.Description = isLevelOneResult.string(forColumn: "DESCRIPTION")
            trackInfo.Id = Int(isLevelOneResult.int(forColumn: "ID"))

            //  print(trackInfo.Description)

            trackArray.add(trackInfo)
        }
    }
    sharedInstance.database!.close()
    return trackArray
}
```

11. Scala

Scala stands for scalable programming language. Its an object oriented, hybrid functional programming language. It has features of an Object Oriented and Functional programming language.
Mainly used for Big Data Analytics.

Example 1 : Print your name

```
object ExPrintName {
        def main (args: Array[String]) {
                println(" My name is John Smith! ")
        }
}
```

Output - My name is John Smith!

Example 2 : Print and Calculate sum of all elements

```
object ExampleArray1  {

        def main (args: Array[String] ) {

        var numbers = Array(10, 20, 30, 40)

        var N:Int = 0;

        // print all array elements
        println ("All array elements :");
        for (N <- numbers)
                {
                        println(N);
                }
// Calculating Sum of all Elements
var sum: Int=0;
for (N <- numbers) {
                sum + = N;
        }
println("Sum of all array elements : " +sum);
        }
}
```

Output - All array elements:

```
10
20
30
40
Sum of all array elements : 100
```