**MoviesApp Documentation**

MoviesApp is a iOS Mobile Application where can see our favorite movies list and movie detail description. This MoviesApp contains three pages like login, movies list and movie details.

**Project Details**:

The MoviesApp project contains three viewControllers named as LoginVC, MoviesVC, DetailMovieVC etc.

**LoginVC:**

**UI Design:**

we have created two UITextFields for username and password details, one Button for login, one UIImageView to show Marvel logo, one UIView to show some Quote. This UIView contains two Labels to show Quote and Name.

@IBOutlet weak var UserTextField: UITextField!

@IBOutlet weak var PasswordTextField: UITextField!

**Code Functionality:**

we created an IBAction function for login button to login to the app. the loginBtn function contains some validation to enter to the MoviesVC by using some if else statements. we can see the loginBtn functionality below

@IBAction func loginBtn(\_ sender: Any) {

if UserTextField.text!.isEmpty{

let alert = UIAlertController(title: "Please Check", message: "Username cannot be empty", preferredStyle: .alert)

alert.addAction(UIAlertAction(title: "ok", style: .default, handler: { (UIAlertAction) in

}))

self.present(alert, animated: true, completion: nil)

}else if UserTextField.text!.characters.count < 6 {

let alert = UIAlertController(title: "Please Check", message: "Enter a Vaild Username, username should be more than 6 letters", preferredStyle: .alert)

alert.addAction(UIAlertAction(title: "ok", style: .default, handler: { (UIAlertAction) in

}))

self.present(alert, animated: true, completion: nil)

}else if PasswordTextField.text!.isEmpty{

let alert = UIAlertController(title: "Please Check", message: "Password cannot be empty", preferredStyle: .alert)

alert.addAction(UIAlertAction(title: "ok", style: .default, handler: { (UIAlertAction) in

}))

self.present(alert, animated: true, completion: nil)

}else if PasswordTextField.text!.characters.count < 6{

let alert = UIAlertController(title: "Plaese Check", message: "Enter a Vaild Password, password should be more than 6 letters", preferredStyle: .alert)

alert.addAction(UIAlertAction(title: "ok", style: .default, handler: { (UIAlertAction) in

}))

self.present(alert, animated: true, completion: nil)

} else if UserTextField.text!.characters.count > 6 , PasswordTextField.text!.characters.count > 6{

performSegue(withIdentifier: "MoviesVC", sender: nil)

}

}

**MoviesVC:**

**UI Design:**

In MovieVC we have created on UIView with a Label show the title of the screen and one UIImageView to show the background image, In UIImageView we have placed a UICollectionView to show the movie list. UICollectionView Contains a UICollectionCell named as moviescollectionviewcell with one UIImageView to show the movie image and one Label to show the movie title. we have placed a Activity Indicator View named as sppiner to progress of the collectionview.

@IBOutlet weak var movieImage: UIImageView!

@IBOutlet weak var movieTitleLbl: UILabel!

@IBOutlet weak var movieImage: UIImageView!

@IBOutlet weak var movieTitleLbl: UILabel!

**Code Fuctionality:**

* In MoviesVC class we have created a function called fetchmovies() to download movies data from the API natively. We can see the funtionality of the fetchmovies() below

func fetchdmovies() {

let url:String = "https://api.themoviedb.org/4/list/1?page=1&api\_key=1fb1e3ef2eb4ae9768fa644588bab3db&sort\_by=title.asc"

let urlRequest = URL(string: url)

let task = URLSession.shared.dataTask(with: urlRequest!) { (data, response, error) in

if (error == nil){

do{

let movielist = try JSONSerialization.jsonObject(with: data!, options: JSONSerialization.ReadingOptions.mutableContainers) as! Dictionary<String,AnyObject>

if let movies = movielist["results"] as? [Dictionary<String,AnyObject>]{

for obj in movies{

let movieList = Movielist(MovieDict: obj)

self.movieslist.append(movieList)

}

DispatchQueue.main.async {

self.collectionView.reloadData()

self.spinner.stopAnimating()

self.spinner.hidesWhenStopped = true

}

}

}catch let err as NSError{

print(err)

}

}else{

print(error.debugDescription)

}

}

task.resume()

}

* In moviescolletionviewcell class we created a function called configureCell() to insert the property value to UILabel and UIImageView. we can see the code below.

func configureCell(movie: Movielist) {

movieTitleLbl.text = movie.movietitle

movieImage.image = UIImage(named: movie.movieImage!)

}

* And to access the functionality of the UICollectionView we need to add the protocols like UICollectionVIewDelegate, UICollectionViewDataSoure etc. To show the actual collectionview which contains movies list, we need to write some protocol methods of UICollectionVIew.

func numberOfSections(in collectionView: UICollectionView) -> Int {

return 1;

}

func collectionView(\_ collectionView: UICollectionView, numberOfItemsInSection section: Int) -> Int {

return movieslist.count

}

func collectionView(\_ collectionView: UICollectionView, cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {

if let cell = collectionView.dequeueReusableCell(withReuseIdentifier: "moviesCell", for: indexPath) as? moviesCollectionViewCell{

let item = movieslist[indexPath.row]

cell.configureCell(movie: item)

return cell

}

return moviesCollectionViewCell()

}

func collectionView(\_ collectionView: UICollectionView, didSelectItemAt indexPath: IndexPath) {

let item = movieslist[indexPath.row]

performSegue(withIdentifier: "movieDetailVC", sender: item)

}

func collectionView(\_ collectionView: UICollectionView, didSelectItemAt indexPath: IndexPath) helps to us show another screen by clicking any of the item.

* And to pass the data of the of MoviesVC to DetailsMovieVC, we call function called

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {

if segue.identifier == "movieDetailVC" {

if let detailVC = segue.destination as? DeatailMovieVC{

if let item = sender as? Movielist{

detailVC.movielist = item

}

}

}

}

After that we call a function performSegue() in didSeclectItemAt indexPath function to send the data to DetailMovieVC by clicking item.

let item = movieslist[indexPath.row]

performSegue(withIdentifier: "movieDetailVC", sender: item)

* spinner:ActivetyIndicatorView! is used to show some animation while downloading the data from the API, once data downloaded to the collectionview spinner stops the animation and hides from collectionview

spinner.startAnimating()

spinner.stopAnimating()

spinner.hidesWhenStopped = true

**DetailsMovieVC:**

**UI Design:**

In DetailMovieVC we created one UIImageView to show the movie image, some Labels to show movie title,

movie description, movie release date and movie rating, one UIView with a Label to show the Screen title. we can see the UI elements below,

@IBOutlet weak var movieName: UILabel!

@IBOutlet weak var descrtiptionLbl: UILabel!

@IBOutlet weak var movieImage: UIImageView!

@IBOutlet weak var releasedateLbl: UILabel!

@IBOutlet weak var ratingLbl: UILabel!

**Code Functionality:**

* To update the data of the DetailsMovieVC we have created a function called updateUI(). By calling the function updateUI() in ViewDidLoad() the screen will update.

func updateUI(){

movieName.text = movielist.movietitle

descrtiptionLbl.text = movielist.moviedescription

movieImage.image = UIImage(named: movielist.movieImage!)

releasedateLbl.text = movielist.release\_date

ratingLbl.text = "\(movielist.rating!)"

}

And we have created two Buttons at left and right of the top UIView. one Button for back to the moviesVC and another Button for logout to LoginVC.

@IBAction func backtologinBtn(\_ sender: Any) {

performSegue(withIdentifier: "logoutVC", sender: nil)

}

@IBAction func backtoMoviesBtn(\_ sender: Any) {

dismiss(animated: true, completion: nil)

}

**Movieslist class:**

This class used for assigning downloaded data from API to different variables like,

var movieImage:String?

var release\_date:String?

var movietitle:String?

var backgroundImage:String?

var moviedescription:String?

var rating:Double?

**Creating API:**

The Restful API created by using of website called https://www.themoviedb.org/. This implemented by using API key, Access token, list id and created API is

"https://api.themoviedb.org/4/list/1?page=1&api\_key=1fb1e3ef2eb4ae9768fa644588bab3db&sort\_by=title.asc"

**Conclusion:**

In MoviesApp by clicking login button of LoginVC , it sends to MoviesVC and by clicking any of the item of MoviesVC, it sends to DetailsMovieVC and shows movie details. Back button of DetailsMovieVC sends to back to the MoviesVC and logout button sends to LoginVC.