Question 1 - What have you learned recently about iOS development? How did you learn it? Has it changed your approach to building apps?

Recently I have worked on App Extension.I am working on Custom Keyboard, which replace the system keyboard.After a user chooses a custom keyboard, it becomes the keyboard for every app the user opens. It has features to use emoji and short cut to all user's favorite application. So he doesn't have to close current application and go to other application. I have been following blogs and online tutorial like **swiftdeveloperblog.com, natashatherobot.com, Udacity.** Doing more practices making new application while adding new features. Comparing iOS with Objective c, iOS is much easier, convenient and reliable when it comes to reusing data, error handling and coding. Even if you don't have background in iOS you can learn from online tutorial or documentation.

Question 2 - Can you talk about a framework that you've used recently (Apple or third-party)? What did you like/dislike about the framework?

Recently I have used Firebase framework in my applications FriendsBook and OnTheMap for user authentication. I have allowed user to login with Facebook and Email. If they don't have account with application than at first time login, the app will automatically create account and saved under firebase. Now they have added awesome

authentication APIs of Twitter, Github with Facebook. I really liked it not using different frameworks for login authentication and same method of handling authentication for more than Facebook, Github, Twitter and even anonymous users. The thing that i dislike is its documentation and tutorial. It takes lot of time to integrate framework into swift application. If it doesn't work than we have to set up manually with building setup.Firebase uses class using Objective-c So, you can not use classes directly into Swift without using bridging-header. Firebase can be used in different ways just downloading SDK or with Cocoapods. If you get any error handling firebase like framework is not found than it is little bit complicated.

Question 3 - Describe how you would construct a Twitter feed application **(here is an example of Udacity's Twitter feed)** that at minimum can display a company's Twitter page. Please include information about any classes/structs that you would use in the app. Which classes/structs would be the model(s), the controller(s), and the view(s)?

Twitter Feed Application has feature of login, making authenticated request to Twitter API, loading tweets, users, likes or images. You need table views to show user information that is relevant to app. Let user compose new tweets as part of app to share content via Twitter Application. Login and table views is a View, User is going to interact with. User information and twitter API are our model or data which will be persistent, user can add new tweet or edit tweet by deleting that.

Controller will be classes, fetching data from View (interface) and handling all methods to manipulate Model(Data).
First we need class or structure to store twitter informations.

```
Class Twitter
{
  var likes: I
  var postTitle: String
  var post summary: String
  var postDate: NSDate
  var postImage: UIImage

// Authenticate user using Twitter API or Firebase API
}
Class TwitterFeed
{
// save the Post in Firebase
}
```

For User to interact
We can provide tableview which can show tweets posted by user like title, summary, picture, Likes
we can allow user to select image from Photo Album or from Camera - if Camera device is not supported than camera will be disabled

I would like to break the app in MVC architecture.
Controllers

LoginController

TwitterFeedController

Model

TwitterClass(Manage Login)

TwitterFeedClass(Manage Posts)

Views

TwitterFeedCells (userName, date ,Title Text, Summary Text, Image, Like Button, Settings)

From MVC it is easy to get idea about core data and structure or classes. To maintain persistent data i would use Core data and manage with NSManagedObject.

Question 4 - Describe some techniques that can be used to ensure that a `UITableView`containing many `UITableViewCell` is displayed at 60 frames per second.

A table view uses cell object to draw its rows visible.cell object is reusable with identifier(dequeueReusableCellWithIdentifier). Tableview asks data source to configure a cell object to display data in cell.At runtime the data source dequeues cells, prepares them and gives them to its table view for drawing the rows. We can not bind data before CellForRowAtIndexpath because cell has to return as quick as possible.Instead of binding data to all tableview cells, Data binding to cell method should be call exactly before showing the cell.

Question 5 - Imagine that you have been given a project that has this

ActorViewController. The`ActorViewController` should be used to display information about an actor. However, to send information to other ViewControllers, it uses NSUserDefaults. Does this make sense to you? How would you send information from one ViewController to another one?

NSUserDefault is used for passing information to keep your data live as long as the app is installed, once the app is removed this will reset automatically for example user parameters so when user opens app, he finds same parameters.There are two other ways to pass values like Segue and Singleton.We can send information making object of View controller in prepareForSegue Method and pass values.Or We can save Actor's information in Structure or Class and use with Singleton. Singleton is basically global object to access variable or function, you have to be very careful when you are mentioning MVC architecture. Thats why I would prefer to use PrepareForSegue, it's a good cause to pass information to another controller.

Question 6 - Imagine that you have been given a project that has this **GithubProjectViewController**. The `GithubProjectViewController` should be used to display high-level information about a GitHub project. However, it's also responsible for finding out if there's network connectivity, connecting to GitHub, parsing the responses and persisting information to disk. It is also one of the biggest classes in the project.

How might you improve the design of this view controller?

For network aspect it would be better to make a separate class to handle network errors and reusability throw out application. Like wise for data handling we need separate class or structure that can be used for persistent information.When you have large amount of code in same class than it is very difficult to understand the logic plus hard to handle any error.
I would break it in 2 different classes.
in View controller class i keep only UI related codes not NS
1) GithubClass- managed data for persistent- add remove or edit
2) GithubAPI- manage Api connection ,parsing, network connectivity

Question 7 - If you were to start your iOS developer position today, what would be your goals a year from now?

I am driven to be the best at what I do and I want to work somewhere where I will have opportunities to develop my skills, take on interesting projects,take on new challenges over time and work with people I can really learn from. Some of the most innovative thinkers in the industry work here and that's a big reason why I would love to build a career here. If I start this position today than I want to grow personally and professionally within this company.In a year I want to see my self improving with knowledge, skills and ability to handle complex projects.