Summary Assignment 1

Human Interface Guidelines

iOS Design Themes
An app design must display these primary themes in his/her app: clarity, deference and depth.

Clarity: The texts and other elements in the detail must be visible
Deference: The content should be aesthetic and constrained to feel the screen for every device
Depth: It should be easy to navigate through the content.

Design Principles
The following principles must be adhered to make the app more marketable: aesthetic integrity, consistency, direct manipulation, feedback, metaphor and user control.

Aesthetic Integrity: The functions of the app should appear and display in a manner that is appealing to the user
Consistency: The app should be presented in a way that people expect
Direct manipulation: Visible results should display when the user manipulated the contents
Feedback: The output of the users actions should be displayed
Metaphor: The app should be programmed in manner that relatable
User Control: The user should have full control of the actions done in the app

App Store Review Guidelines

The guidelines are create to provide the user of the app a safe experience and the developer opportunity to market. The latest guidelines are divided into five sections: Safety, Performance, Business, Design and Legal

Before submitting the application for review, it should meet the checklist provided by the App Store

Safety: The developer should develop the app in a manner that does not cause physical or emotion harm to its user. App created for kids should also follow the kids category guideline. It must follow all the safety guidelines provided by the App Store.

Performance: The app created by a developer should be tested on a device to for bugs before submitting. Beta testing should be done with TestFlight. The user should know what they are getting into when they install the app. Developers must follow the hardware compatibility and software requirements provided by the App Store.

Business: The app should clearly portray its business model. Payments of any kind made in the app should follow the guidelines. The guideline provides do's and don'ts that must be adhered to when implementing business model.

Design: Apps should be designed to be simple and innovative. The functionality should be updated to improve interaction with a user. Developers must come up with new ideas for app and not merely improve on some else's. Apps must have unique features, UI and content that provide lasting entertainment. Developer must ensure not to create multiple Bundle ID for the same app. Apps with extensions must adhere with the extension guideline listed in the App Store. The application must adhere to all the design guidelines.

Legal: Apps must adhere to all legal requirements as well as the local laws. The user's privacy must be protected. Developer can only use content he/she created or is licensed to use. Gambling, gaming and lottery app must follow the regulation provided by App Store. App offering VPN services can only be offered to developers enrolled as an organization. A clear declaration of what user data would be collected and action taken by that data must be made clear prior to the user purchasing the app.

After app and metadata has been submitted for review, the developer should keep in mind: Timing, Status Updates, Expedite Request, Release Date and Rejections.

Ray Wenderlich Swift Style Guide

The guide provide enables the developers code to be readable by ensuring its clear, consistent and concise.

CORRECTNESS
The code should compile without warning

NAMING
The key points provided by Ray Wenderlich that enables the users code to be easy to read and understand.

Prose
The developer must be unambiguous when referring to a method in prose.

Class Prefixes, Delegates, Use Type Inferred Context, Generics and Language should follow the preferred method illustrated

CODE ORGANIZATION
Extensions should be used to display logical blocks of functionality making code easily readable

Protocol Conformance, Unused Code, Minimal imports should follow the preferred format as illustrated.

SPACING
Indent using two space rather than tab to conserve space. Follow the preferred method for spacing .

COMMENTS
Use comments to explain what a particular set of code does.

CLASSES and STRUCT
Classes have reference semantics, therefore should be do things that have identity. Struct have value semantics, therefore should be used to do things that do not have identity.

Use of Self: Avoid using self.

Computed Property: If computed property is read only then omit get clause.

Final: Final should only be used to clarify intent

FUNCTION DECLARATION
Function declaration including opening brace should be on one line. Void should not be used because of lack of input rather use (). Void is used for closure and function outputs.

FUNCTION CALLS
The style of function should be mirrored at call sites

CLOSURE EXPRESSION
Trailing closure syntax should be used only if there is a single closure expression parameter at the end of the argument list.

TYPES
Try to always use swift’s native types and expression .

Constants: Use let instead of var if the value is a constant

Optionals: Variables and functions are declared with ? If nil is acceptable

Lazy Initializer: should be used for finer control over the lifetime of the object

Type Annotation: Use type annotation for empty arrays and dictionaries

Syntactic Sugar: Use shortcut versions of declaration types

FUNCTIONS and METHODS
Free functions which are not attached to class or type should be used sparingly

MEMORY MANAGEMENT
Code should not create reference cycles

Extending Object Lifetime: Object lifetime should be extended similar to the example displayed

ACCESS CONTROL
Full access control is not required

CONTROL FLOW
Use the preferred method provided for the loop control.

Ternary Operator: The ternary operator should only be used when it promotes the codes clarity

GOLDEN PATH
When coding conditionals the left hand side of the code should be the golden/ happy path

Failing Guards: use guard statement to exit in someway

SEMICOLON
Swift does not require the use of semicolon unless multiple statements are written in the same line

PARENTHESES
Parentheses are not required and should therefore be omitted

MULTI-LINE STRING LITERALS
If a string is long use multi- line string literal syntax

NO EMOJI
Do not use emoji