```
// Core Data Model (ChecklistNote.xcdatamodeld)
// Entity: Note
// Attributes: title (String), content (String), isChecked (Boolean)

import CoreData

@objc(Note)
class Note: NSManagedObject {
    @NSManaged var title: String
    @NSManaged var content: String
    @NSManaged var isChecked: Bool
}

// ViewModel - Handling Data
import UIKit
import CoreData

class NotesViewModel {
    private var notes: [Note] = []
    let context = (UIApplication.shared.delegate as!
AppDelegate).persistentContainer.viewContext

    func fetchNotes() {
        let request: NSFetchRequest<Note> = Note.fetchRequest()
        do {
            notes = try context.fetch(request)
        } catch {
            print("Error fetching notes: \(error)")
        }
    }

    func addNote(title: String, content: String, isChecked: Bool) {
        let newNote = Note(context: context)
        newNote.title = title
        newNote.content = content
        newNote.isChecked = isChecked
        saveContext()
    }

    func updateNote(_ note: Note, isChecked: Bool) {
```

```
        note.isChecked = isChecked
        saveContext()
    }

    func deleteNote(_ note: Note) {
        context.delete(note)
        saveContext()
    }

    private func saveContext() {
        do {
            try context.save()
        } catch {
            print("Error saving note: \(error)")
        }
    }

    func getNotes() -> [Note] {
        return notes
    }
}

// View - UI for Notes & Checklist
import UIKit

class NotesViewController: UIViewController, UITableViewDelegate, UITableViewDataSource
{
    let tableView = UITableView()
    var viewModel = NotesViewModel()

    override func viewDidLoad() {
        super.viewDidLoad()
        view.backgroundColor = .white
        setupUI()
        viewModel.fetchNotes()
    }

    func setupUI() {
        navigationItem.rightBarButtonItem = UIBarButtonItem(barButtonSystemItem: .add, target:
self, action: #selector(addNote))
```

```
        tableView.frame = view.bounds
        tableView.delegate = self
        tableView.dataSource = self
        tableView.register(UITableViewCell.self, forCellReuseIdentifier: "cell")
        view.addSubview(tableView)
    }

    @objc func addNote() {
        let alert = UIAlertController(title: "New Note", message: "Enter note details",
preferredStyle: .alert)
        alert.addTextField { $0.placeholder = "Title" }
        alert.addTextField { $0.placeholder = "Content" }

        let addAction = UIAlertAction(title: "Save", style: .default) { _ in
            if let title = alert.textFields?[0].text, let content = alert.textFields?[1].text {
                self.viewModel.addNote(title: title, content: content, isChecked: false)
                self.tableView.reloadData()
            }
        }
        alert.addAction(addAction)
        present(alert, animated: true)
    }

    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return viewModel.getNotes().count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) ->
UITableViewCell {
        let cell = tableView.dequeueReusableCell(withIdentifier: "cell", for: indexPath)
        let note = viewModel.getNotes()[indexPath.row]
        cell.textLabel?.text = note.title
        cell.accessoryType = note.isChecked ? .checkmark : .none
        return cell
    }

    func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
        let note = viewModel.getNotes()[indexPath.row]
        viewModel.updateNote(note, isChecked: !note.isChecked)
        tableView.reloadRows(at: [indexPath], with: .automatic)
```

```
    }
}

// Text Formatting (Bold, Italic, Underline)
import UIKit

class TextFormatter {
    static func formatText(_ text: String, style: UIFont.TextStyle) -> NSAttributedString {
        let attributes: [NSAttributedString.Key: Any] = [.font: UIFont.preferredFont(forTextStyle:
style)]
        return NSAttributedString(string: text, attributes: attributes)
    }
}
```