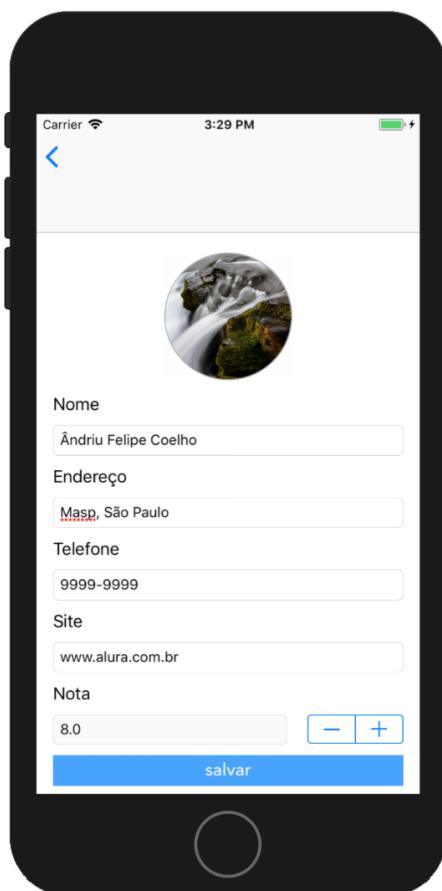


02

Iniciando com Core Data

De volta ao projeto, agora estamos conseguindo tirar foto ou escolher uma a partir da biblioteca do iOS.

Vamos preencher o formulário:



Agora que o formulário está preenchido, como poderíamos salvar esse aluno?

Uma das formas seria criar uma classe Aluno e seus atributos de acordo com os campos do formulário e depois colocá-lo em uma lista.

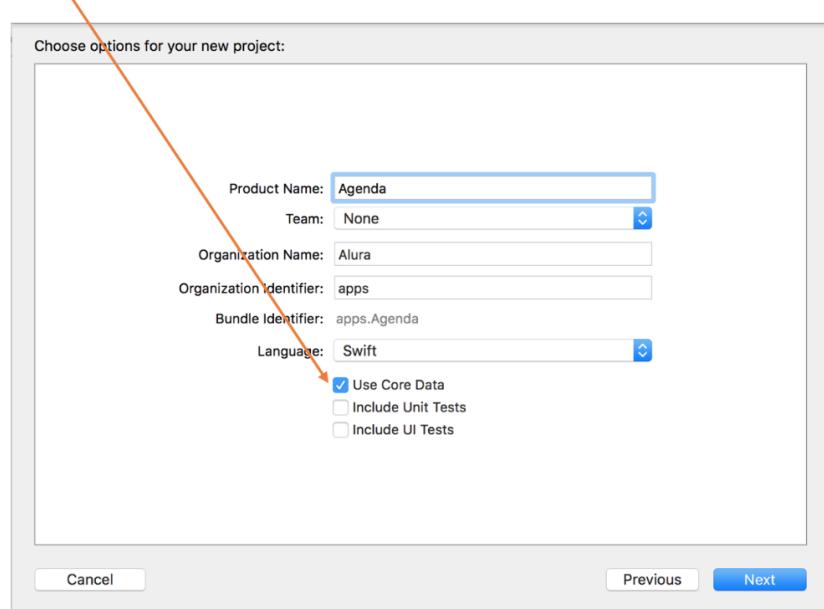
O problema dessa abordagem é que toda vez que o app for encerrado, como as variáveis estão alocadas em memória, serão apagadas.

Para conseguir salvar os objetos, veremos agora como trabalhar com um framework nativo do iOS que se chama **Core Data**.

O Core Data é um framework nativo do iOS que permite persistir informações de forma orientada a objetos. Um das vantagens é que não precisamos digitar comandos SQL, pois o próprio core data possui uma coleção de objetos (**Core Data Stack**) que gerencia a manipulação dos objetos a serem persistidos.

Para utilizar o CoreData é importante setar essa configuração quando estamos **criando o projeto**:

opção do Core Data
marcada no início do projeto



Agora podemos abrir o arquivo `AppDelegate.swift` para conferir as implementações do Core Data que já vem no nosso projeto:

arquivo `AppDelegate` aberto

```

// MARK: - Core Data stack
lazy var persistentContainer: NSPersistentContainer = {
    /*
        The persistent container for the application. This implementation
        creates and returns a container, having loaded the store for the
        application to it. This property is optional since there are legitimate
        error conditions that could cause the creation of the store to fail.
    */
    let container = NSPersistentContainer(name: "Agenda")
    container.loadPersistentStores(completionHandler: { (storeDescription, error) in
        if let error = error as NSError? {
            // Replace this implementation with code to handle the error appropriately.
            // fatalError() causes the application to generate a crash log and terminate.
            // You should not use this function in a shipping application, although it may be useful during development.
            /*
            Typical reasons for an error here include:
            * The parent directory does not exist, cannot be created, or disallows writing.
            * The persistent store is not accessible, due to permissions or data protection when the device is locked.
            * The device is out of space.
            * The store could not be migrated to the current model version.
            Check the error message to determine what the actual problem was.
            */
            fatalError("Unresolved error \(error), \(error.userInfo)")
        }
    })
    return container
}()

// MARK: - Core Data Saving support

func saveContext () {
    let context = persistentContainer.viewContext
    if context.hasChanges {
        do {
            try context.save()
        } catch {
            // Replace this implementation with code to handle the error appropriately.
            // fatalError() causes the application to generate a crash log and terminate.
            // You should not use this function in a shipping application, although it may be useful during development.
            let nserror = error as NSError
            fatalError("Unresolved error \(nserror), \(nserror.userInfo)")
        }
    }
}

```

implementações do Core Data

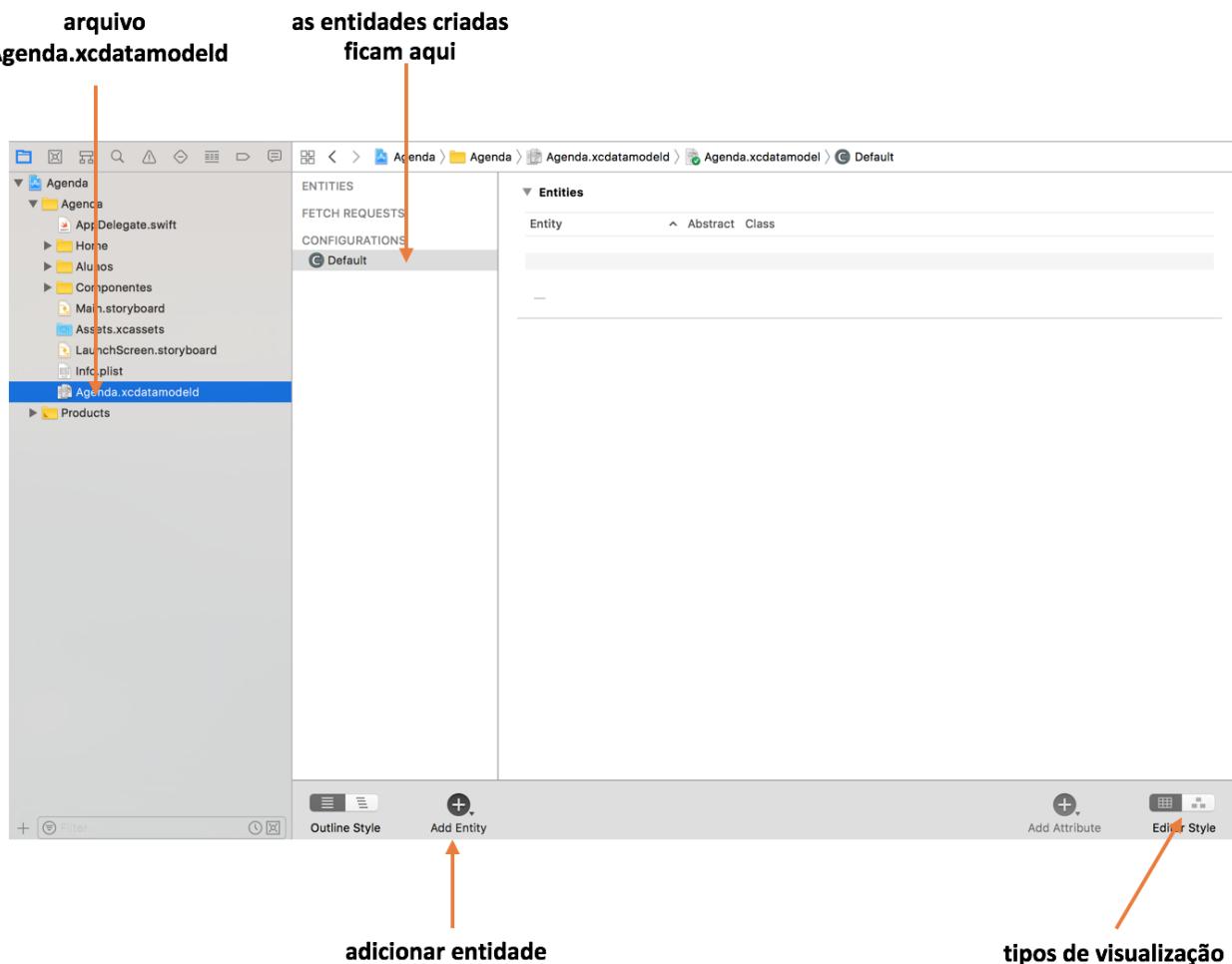
Aqui nós temos a implementação do **Persistent Container** do Core Data que é o responsável por fazer todas as operações como:

Managed Object Model: Definição do Schema dos objetos que serão salvos no banco, ou seja, quais atributos (nome, endereço, imagem...) e quais tipos (String, Int, Bool) serão salvos nessa entidade.

Managed Object Context: Faz operações de manipulação dos objetos como: Create, Read, Update, Delete (CRUD).

Persistent Store Coordinator: É responsável pelo acesso e persistência dos objetos no banco.

Então vamos começar a trabalhar com o Core Data. O primeiro passo é abrirmos o arquivo **Agenda.xcdatamodeld**



É através desse arquivo que vamos criar entidades, configurar relacionamentos, definir Schema, ou seja, todas as configurações necessárias para o Core Data criar o banco de dados do nosso aplicativo.

Agora é a sua vez:

- Crie a entidade **Aluno** e seus atributos de acordo com os campos do formulário.