# Just in time consent

Just in time (JIT) is a widget within My account, which provides an application’s developer, functionality to set a specific consent at the point where it is needed.

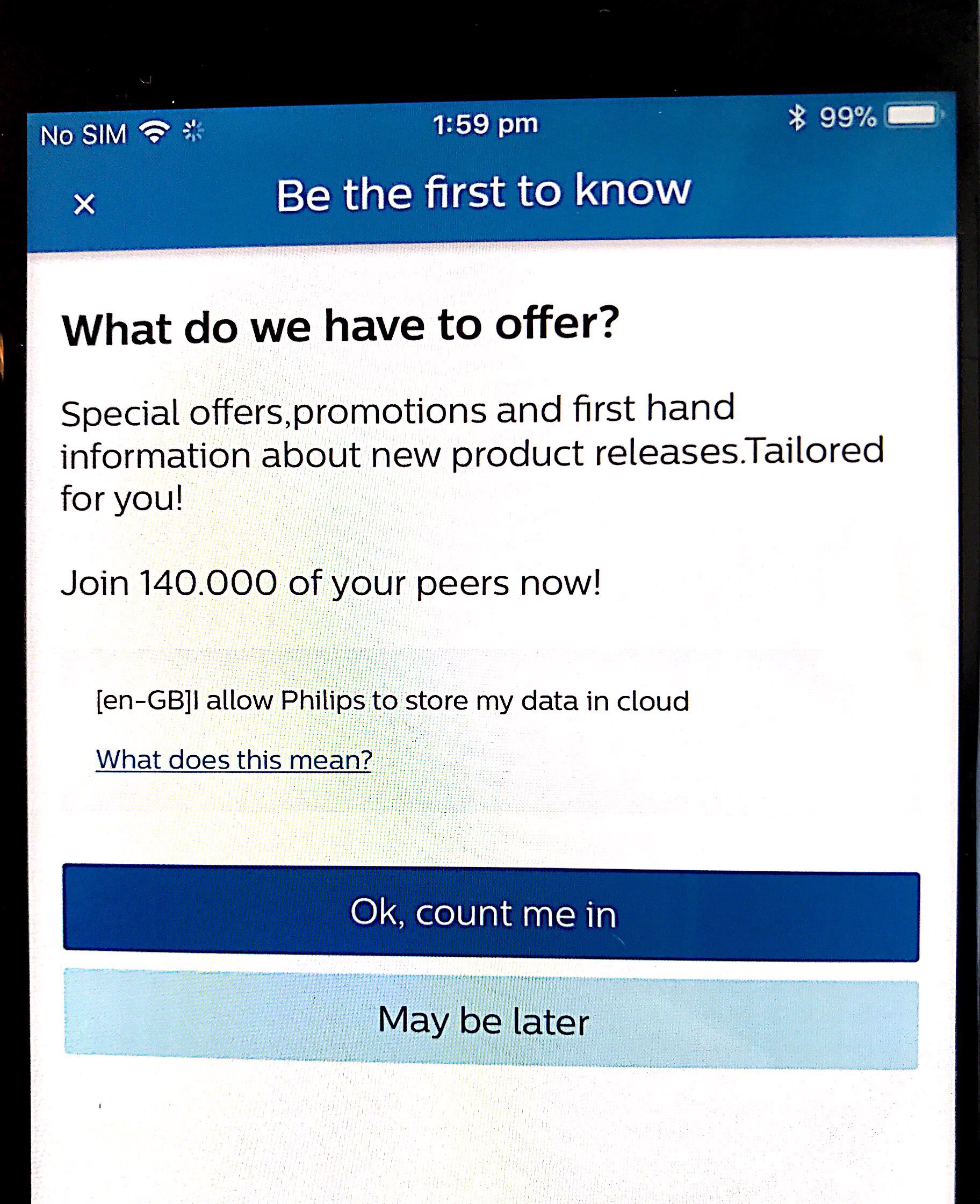


Figure 1: Just in time (JIT) screen

## Launching JIT screen

 guard consentDefenitionState?.status == ConsentStatus.active ||

                      consentDefenitionState?.status == ConsentStatus.rejected  else{

                    self?.launchJustInTimeConsentScreen()

                    return

                }

## JustInTimeConsentViewProtocol Implementation example

extension ChapterDetailsViewController:JustInTimeConsentViewProtocol{

    func launchJustInTimeConsentScreen(){

        let consentDefinition = getConsentDefinitionForMomentType()

        let justInTimeUIConfig = JustInTimeUIConfig(title: "Be the first to know",

acceptButtonTitle: "Ok, count me in",

cancelButtonTitle: "May be later",

userBenefitsTitle: "What do we have to offer?",

userBenefitsDescription: "Special offers,promotions and first hand information about new product releases.Tailored for you! \n\nJoin 140.000 of your peers now!")

        let justInTimeConsentInterface = JustInTimeConsentInterface(withConsentDefinition: consentDefinition, withConsentInterface: getConsentInteractor(), withUIConfig: justInTimeUIConfig,withAppInfra:appinfra)

        let justInTimeConsentVC = justInTimeConsentInterface.getJustInTimeConsentViewController(justInTimeConsentDelegate: self)

        let navigationController = UINavigationController(rootViewController: justInTimeConsentVC)

        DispatchQueue.main.async {

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

        }

    }

    private func getConsentInteractor() -> ConsentInteractor{

        let communicator = ConsentServiceNetworkCommunicator()

        let consentsClient = ConsentsClient(communicator: communicator, appInfra: appinfra)

        return ConsentInteractor(consentsClient: consentsClient, appConfig: appinfra.appConfig, appInfra: appinfra)

    }

    private func getConsentDefinitionForMomentType() -> ConsentDefinition{

        return(ConsentBootstrapper.sharedInstance.consentConfiguration.getConsentDefinitionHandlerMappingForType(type:"moment")?.getConsentDefinition())!

    }

    func getMomentConsentStatus(completion: @escaping (\_ consentDefenitionState: ConsentDefinitionStatus?, \_error: NSError?) -> Void){

        getConsentInteractor().fetchConsentState(forConsentDefinition: getConsentDefinitionForMomentType()) { (consentDefenitionState, error) in

            completion(consentDefenitionState,error)

        }

    }

// JustInTimeConsentViewProtocol methods

    func justInTimeConsentAccepted() {

        print("User has accepted the consent")

        DispatchQueue.main.async {

            self.dismiss(animated: false, completion: nil)

            \_ = self.presenter?.onEvent(self.testDataServicesEventId)

        }

    }

    func justInTimeConsentCancelled() {

        print("User has cancelled the consent")

        DispatchQueue.main.async {

            self.dismiss(animated: false, completion: nil)

            \_ = self.presenter?.onEvent(self.testDataServicesEventId)

        }

    }

    func justInTimeConsentDismissed(){

        print("User has dismissed the consent")

        DispatchQueue.main.async {

            self.dismissNav()

        }

    }

}

# 

## Presenting the JIT screen modally

## To present the JIT screen modally, we can use the following

/\*use this to present JIT screen modally\*/

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

/\*use this to push JIT screen\*/

self.navigationController?.pushViewController(justInTimeConsentVC, animated: true)

## Implement the following Protocol

@objc public protocol JustInTimeConsentViewProtocol:NSObjectProtocol{

func justInTimeConsentAccepted()

func justInTimeConsentCancelled()

@objc optional func justInTimeConsentDismissed()

}

## Public Interface as a whole

public class JustInTimeUIConfig: NSObject {

public init(inTitle:String,inAcceptButtonTite:String,inCancelButtonTitle:String)

public convenience init(title:String,acceptButtonTitle:String,cancelButtonTitle:String,userBenefitsTitle:String,userBenefitsDescription:String)

}

public class JustInTimeConsentInterface: NSObject {

public init(withConsentDefinition: ConsentDefinition, withConsentInterface: ConsentInteractorProtocol, withUIConfig:JustInTimeUIConfig, withAppInfra:AIAppInfra)

public func getJustInTimeConsentViewController(justInTimeConsentDelegate: JustInTimeConsentViewProtocol) -> UIViewController

}