#WWDC19

# Subscription Offers Best Practices

Ross LeBeau, App Store Engineer
Michael Gargas, Technical Advocate, App Store Commerce

# Overview

# Overview

Discounted price for specific duration

# Overview

Discounted price for specific duration

Powerful tool for mitigating churn

# Overview

Discounted price for specific duration

Powerful tool for mitigating churn

You choose when to offer to customers

# Overview

Discounted price for specific duration

Powerful tool for mitigating churn

You choose when to offer to customers

Customers may redeem as many offers as you allow

# Overview

Discounted price for specific duration

Powerful tool for mitigating churn

You choose when to offer to customers

Customers may redeem as many offers as you allow

For existing or previously subscribed customers

Setup

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

Using offers to reduce churn

Setup

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

Using offers to reduce churn

Setup

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

Using offers to reduce churn

Setup

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

Using offers to reduce churn

Setup

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

Using offers to reduce churn

Setup

Generating a signature

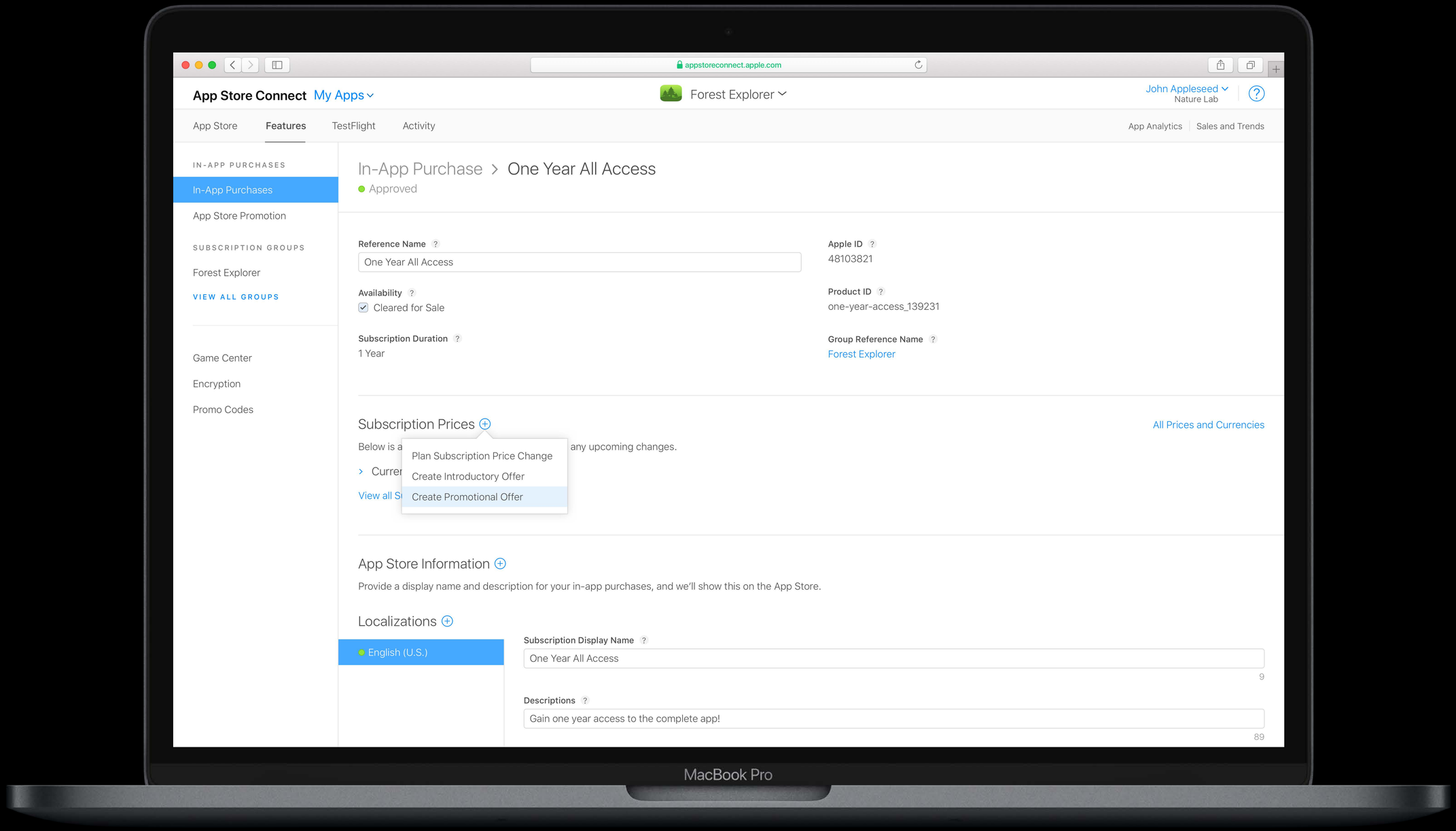
Interacting with the StoreKit API

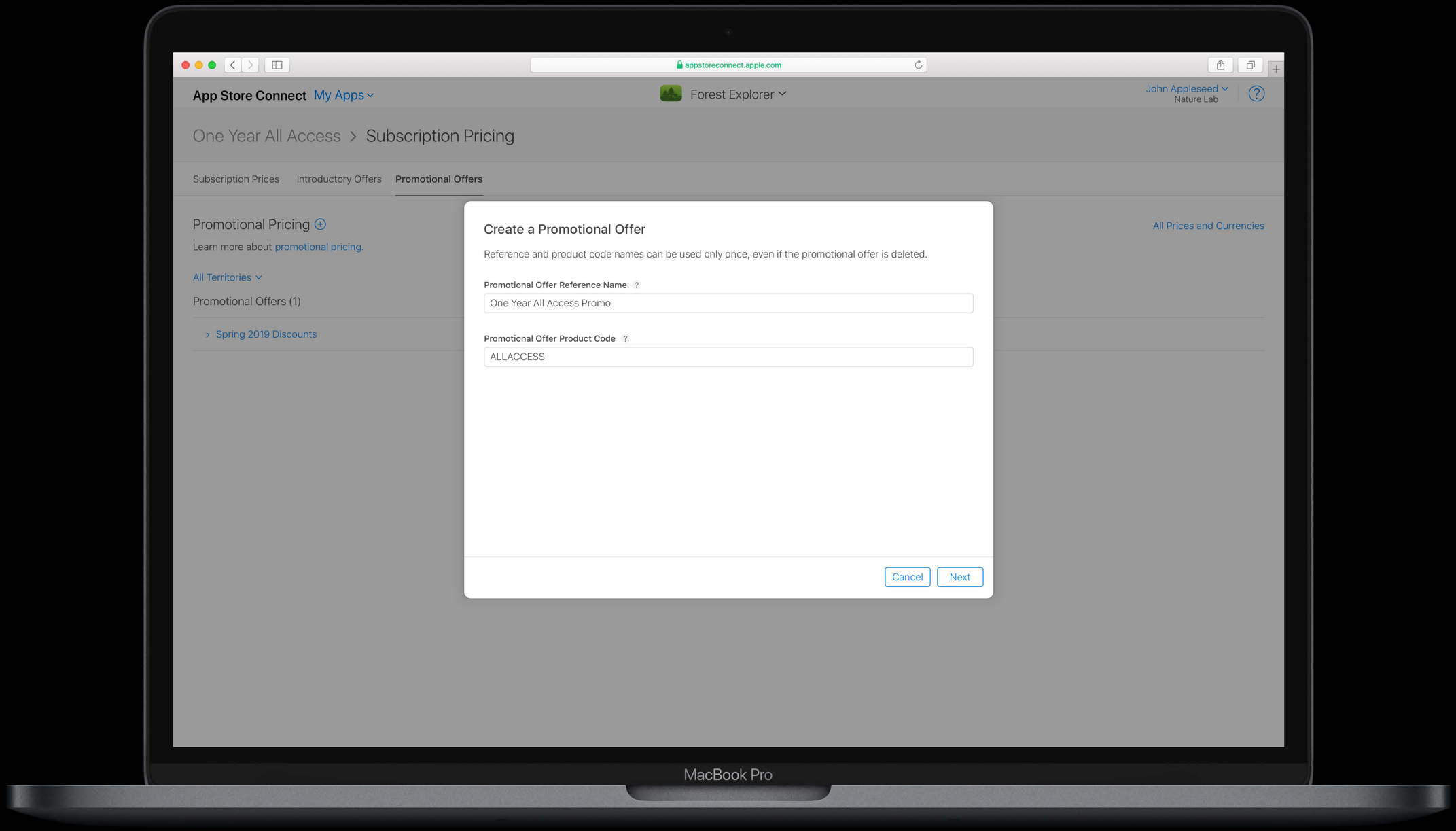
Determining customer eligibility

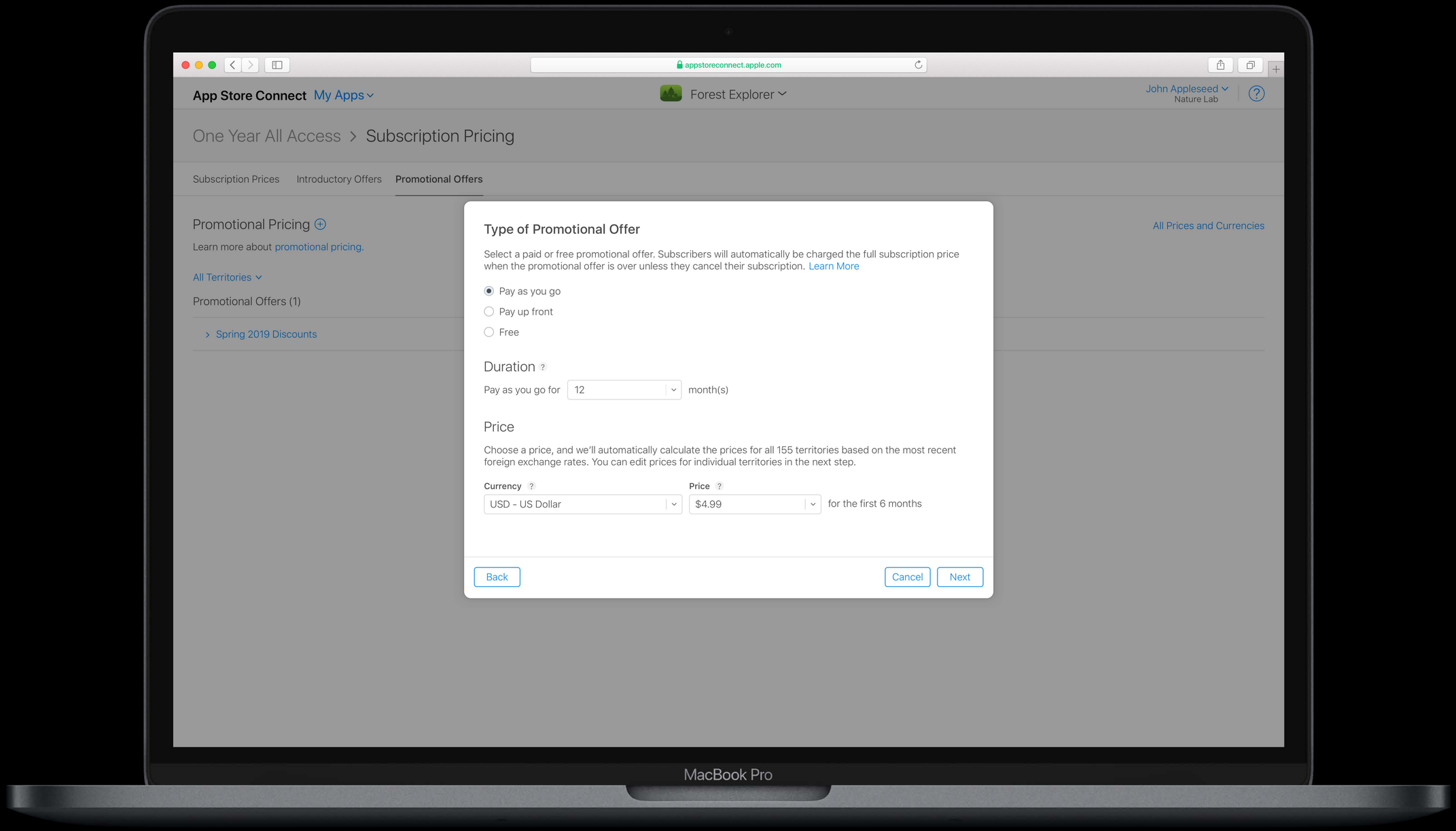
Distributing offers

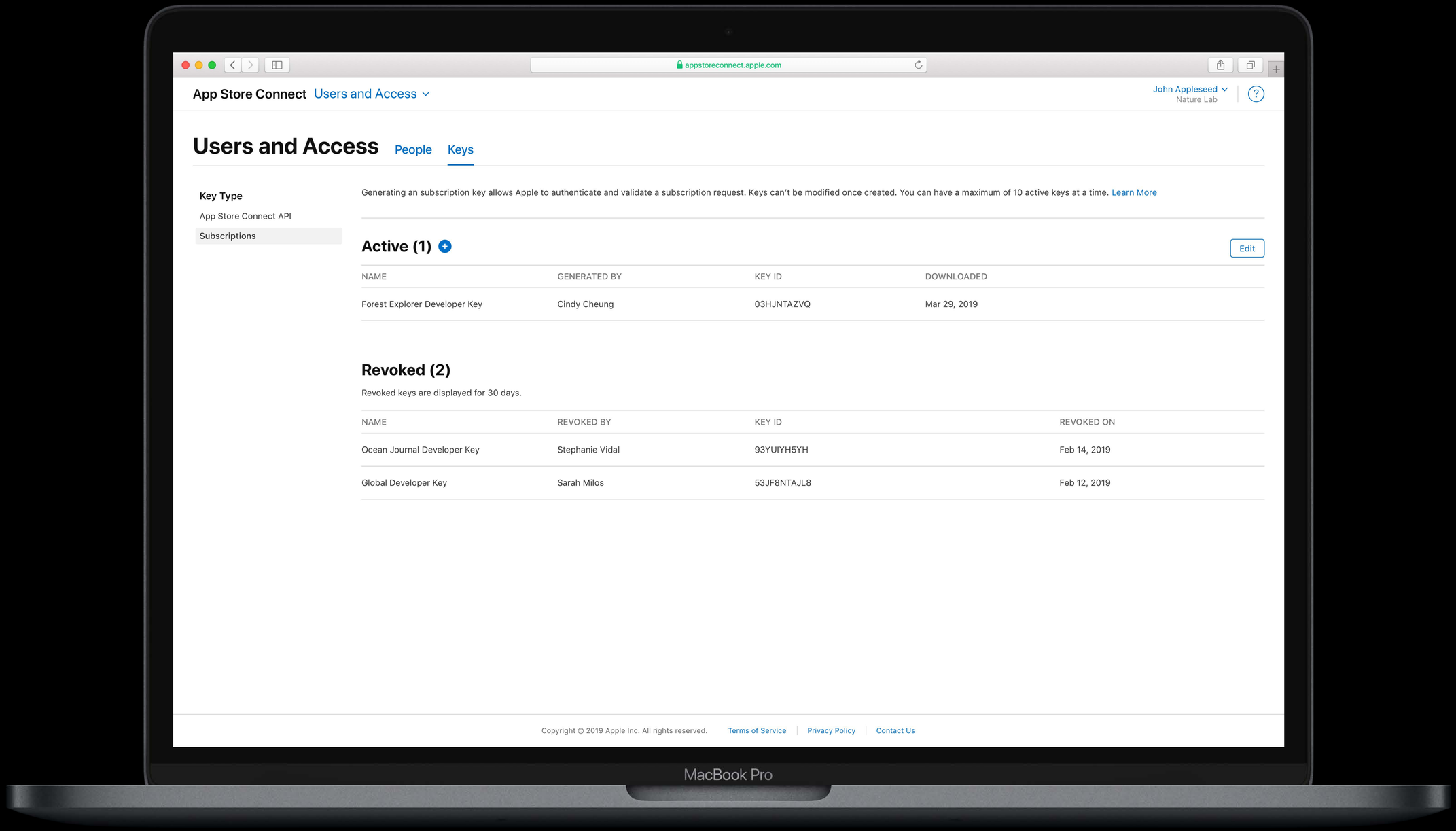
Using offers to reduce churn

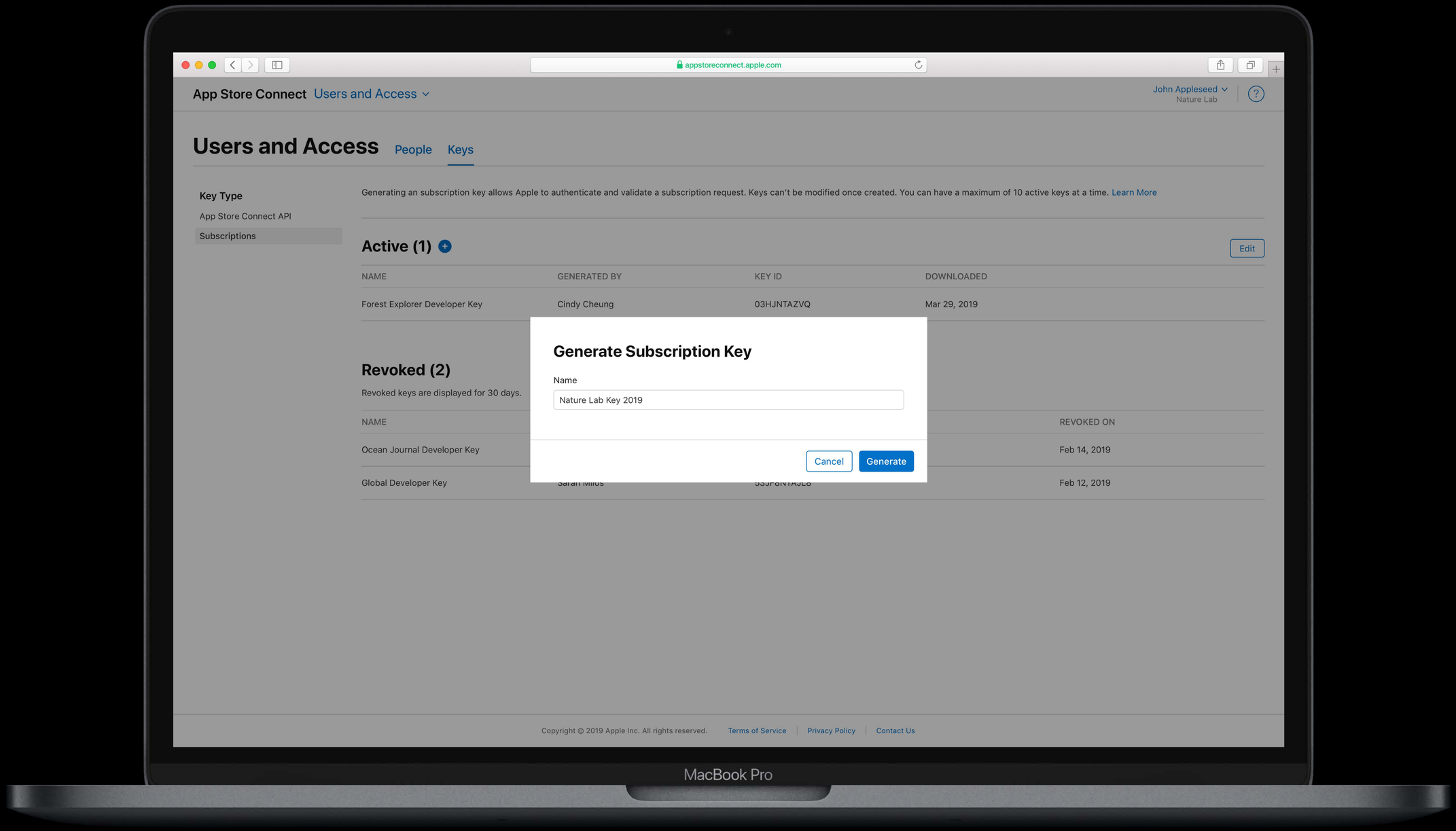
# Setup

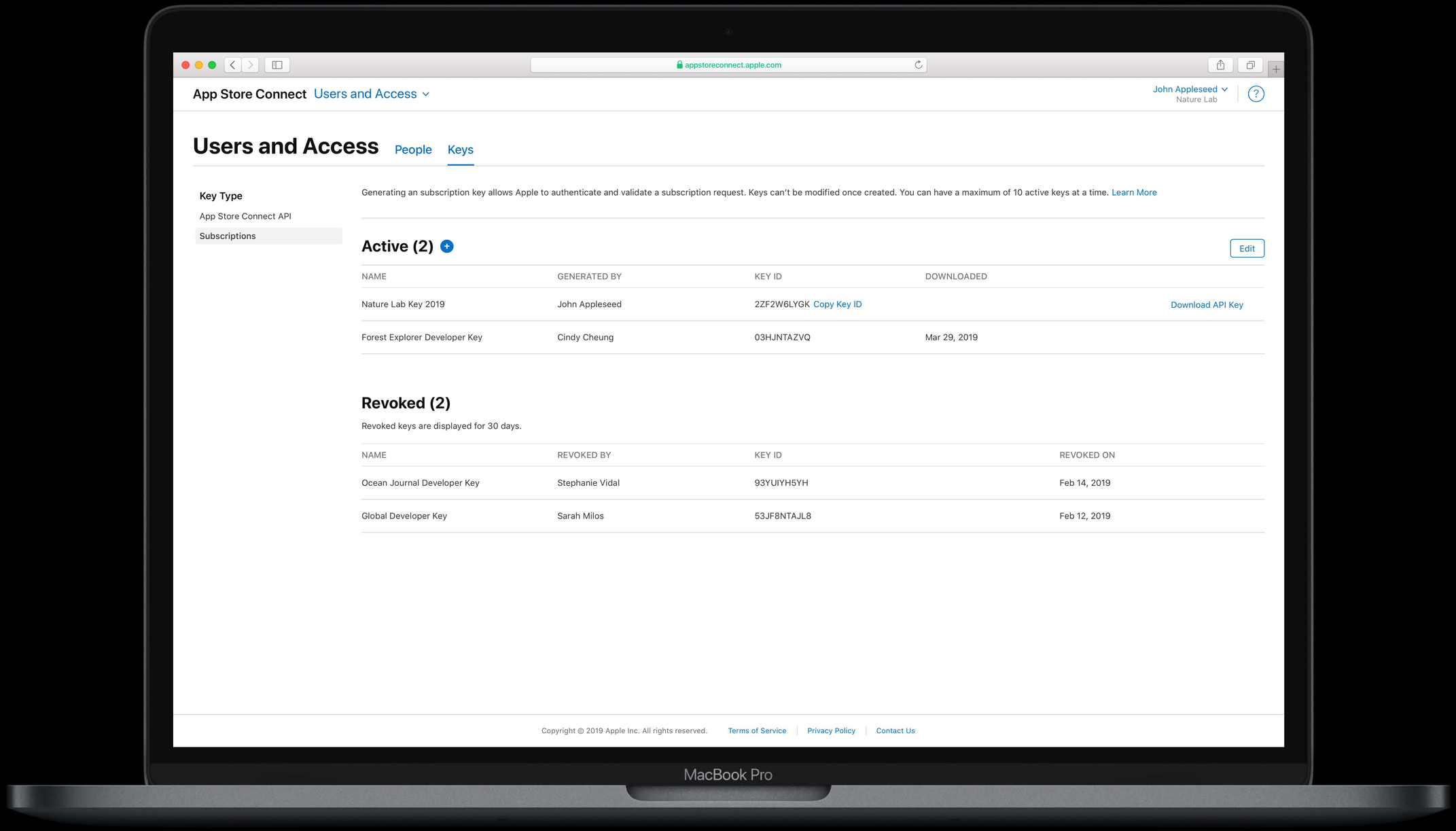

appstoreconnect.apple.com
App Store Connect
My Apps
Forest Explorer
John Appleseed
Nature Lab
App Store
Features
TestFlight
Activity
App Analytics
Sales and Trends
IN-APP PURCHASES
In-App Purchases
App Store Promotion
SUBSCRIPTION GROUPS
Forest Explorer
VIEW ALL GROUPS
Game Center
Encryption
Promo Codes
In-App Purchase > One Year All Access
Approved
Reference Name
One Year All Access
Apple ID
48103821
Availability
Cleared for Sale
Product ID
one-year-access_139231
Subscription Duration
1 Year
Group Reference Name
Forest Explorer
Subscription Prices
All Prices and Currencies
Plan Subscription Price Change
Create Introductory Offer
Create Promotional Offer
any upcoming changes.
App Store Information
Provide a display name and description for your in-app purchases, and we'll show this on the App Store.
Localizations
English (U.S.)
Subscription Display Name
One Year All Access
9
Descriptions
Gain one year access to the complete app!
89
MacBook Pro


appstoreconnect.apple.com
App Store Connect My Apps
Forest Explorer
John Appleseed
Nature Lab
One Year All Access > Subscription Pricing
Subscription Prices
Introductory Offers
Promotional Offers
Promotional Pricing
Learn more about promotional pricing.
All Territories
Promotional Offers (1)
Spring 2019 Discounts
All Prices and Currencies
Create a Promotional Offer
Reference and product code names can be used only once, even if the promotional offer is deleted.
Promotional Offer Reference Name
One Year All Access Promo
Promotional Offer Product Code
ALLACCESS
Cancel
Next
MacBook Pro


appstoreconnect.apple.com
App Store Connect
My Apps
Forest Explorer
John Appleseed
Nature Lab
One Year All Access
Subscription Pricing
Subscription Prices
Introductory Offers
Promotional Offers
Promotional Pricing
Learn more about promotional pricing.
All Territories
Promotional Offers (1)
Spring 2019 Discounts
All Prices and Currencies
Type of Promotional Offer
Select a paid or free promotional offer. Subscribers will automatically be charged the full subscription price when the promotional offer is over unless they cancel their subscription. Learn More
Pay as you go
Pay up front
Free
Duration
Pay as you go for
12
month(s)
Price
Choose a price, and we'll automatically calculate the prices for all 155 territories based on the most recent foreign exchange rates. You can edit prices for individual territories in the next step.
Currency
Price
USD - US Dollar
$4.99
for the first 6 months
Back
Cancel
Next
MacBook Pro


appstoreconnect.apple.com
App Store Connect Users and Access
John Appleseed
Nature Lab
Users and Access
People
Keys
Key Type
App Store Connect API
Subscriptions
Generating an subscription key allows Apple to authenticate and validate a subscription request. Keys can't be modified once created. You can have a maximum of 10 active keys at a time. Learn More
Active (1)
Edit
NAME
GENERATED BY
KEY ID
DOWNLOADED
Forest Explorer Developer Key
Cindy Cheung
03HJNTAZVQ
Mar 29, 2019
Revoked (2)
Revoked keys are displayed for 30 days.
NAME
REVOKED BY
KEY ID
REVOKED ON
Ocean Journal Developer Key
Stephanie Vidal
93YUIYH5YH
Feb 14, 2019
Global Developer Key
Sarah Milos
53JF8NTAJL8
Feb 12, 2019
Copyright © 2019 Apple Inc. All rights reserved.
Terms of Service
Privacy Policy
Contact Us
MacBook Pro


appstoreconnect.apple.com
App Store Connect Users and Access
John Appleseed
Nature Lab
Users and Access
People
Keys
Key Type
App Store Connect API
Subscriptions
Generating an subscription key allows Apple to authenticate and validate a subscription request. Keys can't be modified once created. You can have a maximum of 10 active keys at a time. Learn More
Active (1)
Edit
NAME
GENERATED BY
KEY ID
DOWNLOADED
Forest Explorer Developer Key
Cindy Cheung
03HJNTAZVQ
Mar 29, 2019
Generate Subscription Key
Name
Nature Lab Key 2019
Cancel
Generate
Revoked (2)
Revoked keys are displayed for 30 days.
NAME
REVOKED ON
Ocean Journal Developer Key
Feb 14, 2019
Global Developer Key
Feb 12, 2019
Copyright © 2019 Apple Inc. All rights reserved.
Terms of Service
Privacy Policy
Contact Us
MacBook Pro


appstoreconnect.apple.com
App Store Connect Users and Access
John Appleseed
Nature Lab
Users and Access People Keys
Key Type
App Store Connect API
Subscriptions
Generating an subscription key allows Apple to authenticate and validate a subscription request. Keys can't be modified once created. You can have a maximum of 10 active keys at a time. Learn More
Active (2)
Edit
NAME GENERATED BY KEY ID DOWNLOADED
Nature Lab Key 2019 John Appleseed 2ZF2W6LYGK Copy Key ID Download API Key
Forest Explorer Developer Key Cindy Cheung 03HJNTAZVQ Mar 29, 2019
Revoked (2)
Revoked keys are displayed for 30 days.
NAME REVOKED BY KEY ID REVOKED ON
Ocean Journal Developer Key Stephanie Vidal 93YUIYH5YH Feb 14, 2019
Global Developer Key Sarah Milos 53JF8NTAJL8 Feb 12, 2019
Copyright © 2019 Apple Inc. All rights reserved. Terms of Service Privacy Policy Contact Us
MacBook Pro

# Subscription Offers Keys

# Subscription Offers Keys

Valid across entire developer account

- All apps
- All offers

Multiple keys can be active at once

# Generating a Signature

# Cryptographic Signature

# Cryptographic Signature

Ensures only authorized users can redeem offers

# Cryptographic Signature

Ensures only authorized users can redeem offers

Payload and signature

# Cryptographic Signature

Ensures only authorized users can redeem offers

Payload and signature

Asymmetric cryptography

# Cryptographic Signature

Ensures only authorized users can redeem offers

Payload and signature

Asymmetric cryptography

- Private key used to sign payload

# Cryptographic Signature

Ensures only authorized users can redeem offers

Payload and signature

Asymmetric cryptography

- Private key used to sign payload
- Public key used to validate signature

```
//Getting the payload data

router.get('/offer', function(req, res) {
    const appBundleID = req.body.appBundleID;
    const productIdentifier = req.body.productIdentifier;
    const subscriptionOfferID = req.body.offerID;
    const applicationUsername = req.body.applicationUsername;

    const nonce = uuidv4();

    const currentDate = new Date();
    const timestamp = currentDate.getTime();

    // ...
});
```

```
//Getting the payload data

router.get('/offer', function(req, res) {
    const appBundleID = req.body.appBundleID;
    const productIdentifier = req.body.productIdentifier;
    const subscriptionOfferID = req.body.offerID;
    const applicationUsername = req.body.applicationUsername;

    const nonce = uuidv4();

    const currentDate = new Date();
    const timestamp = currentDate.getTime();

    // ...
});
```

```
//Getting the payload data

router.get('/offer', function(req, res) {
    const appBundleID = req.body.appBundleID;
    const productIdentifier = req.body.productIdentifier;
    const subscriptionOfferID = req.body.offerID;
    const applicationUsername = req.body.applicationUsername;

    const nonce = uuidv4();

    const currentDate = new Date();
    const timestamp = currentDate.getTime();

    // ...
});
```

```
//Getting the payload data

router.get('/offer', function(req, res) {
    const appBundleID = req.body.appBundleID;
    const productIdentifier = req.body.productIdentifier;
    const subscriptionOfferID = req.body.offerID;
    const applicationUsername = req.body.applicationUsername;

    const nonce = uuidv4();

    const currentDate = new Date();
    const timestamp = currentDate.getTime();

    // ...
});
```

```
//Getting the key ID

router.get('/offer', function(req, res) {
    // ...

    const keyID = getKeyID();

    // ...
});

function getKeyID() {
    // Implement key selection logic here
    return process.env.SUBSCRIPTION_OFFERS_KEY_ID;
}
```

```
//Getting the key ID

router.get('/offer', function(req, res) {
    // ...

    const keyID = getKeyID();

    // ...
});

function getKeyID() {
    // Implement key selection logic here
    return process.env.SUBSCRIPTION_OFFERS_KEY_ID;
}
```

```
//Getting the key ID

router.get('/offer', function(req, res) {
    // ...

    const keyID = getKeyID();

    // ...
});

function getKeyID() {
    // Implement key selection logic here
    return process.env.SUBSCRIPTION_OFFERS_KEY_ID;
}
```

```
//Creating the payload string

router.get('/offer', function(req, res) {
    // ...

    const payload = appBundleID + '\u2063' +
                    keyID + '\u2063' +
                    productIdentifier + '\u2063' +
                    subscriptionOfferID + '\u2063' +
                    applicationUsername  + '\u2063' +
                    nonce + '\u2063' +
                    timestamp;
});
```

```
//Getting the private key

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    // ...
});

function getKeyStringForID(keyID) {
    if (keyID === process.env.SUBSCRIPTION_OFFERS_KEY_ID) {
        return process.env.SUBSCRIPTION_OFFERS_PRIVATE_KEY;
    } else {
        throw "Key ID not recognized";
    }
}
```

```
//Getting the private key

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    // ...
});

function getKeyStringForID(keyID) {
    if (keyID === process.env.SUBSCRIPTION_OFFERS_KEY_ID) {
        return process.env.SUBSCRIPTION_OFFERS_PRIVATE_KEY;
    } else {
        throw "Key ID not recognized";
    }
}
```

```
//Getting the private key

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    // ...
});

function getKeyStringForID(keyID) {
    if (keyID === process.env.SUBSCRIPTION_OFFERS_KEY_ID) {
        return process.env.SUBSCRIPTION_OFFERS_PRIVATE_KEY;
    } else {
        throw "Key ID not recognized";
    }
}
```

```
//Generating the signature

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    const key = new ECKey(keyString, 'pem');

    const cryptoSign = key.createSign('SHA256');

    cryptoSign.update(payload);

    const signature = cryptoSign.sign('base64');
});
```

```
//Generating the signature

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    const key = new ECKey(keyString, 'pem');

    const cryptoSign = key.createSign('SHA256');

    cryptoSign.update(payload);

    const signature = cryptoSign.sign('base64');
});
```

```
//Generating the signature

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    const key = new ECKey(keyString, 'pem');

    const cryptoSign = key.createSign('SHA256');

    cryptoSign.update(payload);

    const signature = cryptoSign.sign('base64');
});
```

```
//Generating the signature

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    const key = new ECKey(keyString, 'pem');

    const cryptoSign = key.createSign('SHA256');

    cryptoSign.update(payload);

    const signature = cryptoSign.sign('base64');
});
```

```
//Generating the signature

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    const key = new ECKey(keyString, 'pem');

    const cryptoSign = key.createSign('SHA256');

    cryptoSign.update(payload);

    const signature = cryptoSign.sign('base64');
});
```

```
//Verifying the signature and sending the response

router.get('/offer', function(req, res) {
    // ...

    const verificationResult = key.createVerify('SHA256').update(payload).verify(signature,
        'base64');
    console.log("Verification result: " + verificationResult)

    res.setHeader('Content-Type', 'application/json');
    res.json({
        'keyID': keyID,
        'nonce': nonce,
        'timestamp': timestamp,
        'signature': signature
    });
});
```

```
//Verifying the signature and sending the response

router.get('/offer', function(req, res) {
    // ...

    const verificationResult = key.createVerify('SHA256').update(payload).verify(signature,
        'base64');
    console.log("Verification result: " + verificationResult)

    res.setHeader('Content-Type', 'application/json');
    res.json({
        'keyID': keyID,
        'nonce': nonce,
        'timestamp': timestamp,
        'signature': signature
    });
});
```

```
//Verifying the signature and sending the response

router.get('/offer', function(req, res) {
    // ...

    const verificationResult = key.createVerify('SHA256').update(payload).verify(signature,
        'base64');
    console.log("Verification result: " + verificationResult)

    res.setHeader('Content-Type', 'application/json');
    res.json({
        'keyID': keyID,
        'nonce': nonce,
        'timestamp': timestamp,
        'signature': signature
    });
});
```

# Interacting with the StoreKit API

# Interacting with the StoreKit API

# Interacting with the StoreKit API

Get offer details from the App Store

# Interacting with the StoreKit API

Get offer details from the App Store

Send signature request to server

# Interacting with the StoreKit API

Get offer details from the App Store

Send signature request to server

- Send request when customer is ready to purchase

# Interacting with the StoreKit API

Get offer details from the App Store

Send signature request to server

- Send request when customer is ready to purchase
- E.g. right before displaying your offer or store UI

# Interacting with the StoreKit API

Get offer details from the App Store

Send signature request to server

- Send request when customer is ready to purchase
- E.g. right before displaying your offer or store UI
- This ensures recent timestamp and up-to-date business logic

# Interacting with the StoreKit API

Get offer details from the App Store

Send signature request to server

- Send request when customer is ready to purchase
- E.g. right before displaying your offer or store UI
- This ensures recent timestamp and up-to-date business logic

Handle signature response from server

```
//Getting Offer Details

open class SKProduct : NSObject {
    open var discounts: [SKProductDiscount] { get }
}
```

```
//Getting Offer Details

open class SKProduct : NSObject {
    open var discounts: [SKProductDiscount] { get }
}

open class SKProductDiscount : NSObject {
    open var price: NSDecimalNumber { get }
    open var priceLocale: Locale { get }
    open var identifier: String? { get }
    open var subscriptionPeriod: SKProductSubscriptionPeriod { get }
    open var numberOfPeriods: Int { get }
    open var paymentMode: SKProductDiscount.PaymentMode { get }
    open var type: SKProductDiscount.Type { get }
}
```

```
//Getting Offer Details

open class SKProduct : NSObject {
    open var discounts: [SKProductDiscount] { get }
}

open class SKProductDiscount : NSObject {
    open var price: NSDecimalNumber { get }
    open var priceLocale: Locale { get }
    open var identifier: String? { get }
    open var subscriptionPeriod: SKProductSubscriptionPeriod { get }
    open var numberOfPeriods: Int { get }
    open var paymentMode: SKProductDiscount.PaymentMode { get }
    open var type: SKProductDiscount.Type { get }
}
```

```
//Getting Offer Details

open class SKProduct : NSObject {
    open var discounts: [SKProductDiscount] { get }
}

open class SKProductDiscount : NSObject {
    open var price: NSDecimalNumber { get }
    open var priceLocale: Locale { get }
    open var identifier: String? { get }
    open var subscriptionPeriod: SKProductSubscriptionPeriod { get }
    open var numberOfPeriods: Int { get }
    open var paymentMode: SKProductDiscount.PaymentMode { get }
    open var type: SKProductDiscount.Type { get }
}
```

```
//Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier:
String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier,
offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber,
keyIdentifier: String, signature: String) in

        // Handle response
    })
}
```

```
//Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier:
String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier,
offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber,
keyIdentifier: String, signature: String) in

        // Handle response
    })
}
```

```
//Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier:
String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier,
offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber,
keyIdentifier: String, signature: String) in

        // Handle response
    })
}
```

```
//Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier:
String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier,
offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber,
keyIdentifier: String, signature: String) in

        // Handle response
    })
}
```

```
//Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier:
String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier,
offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber,
keyIdentifier: String, signature: String) in

        // Handle response
    })
}
```

```
//Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier:
String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier,
offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber,
keyIdentifier: String, signature: String) in

        // Handle response
    })
}
```

```
//Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier:
String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier,
offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber,
keyIdentifier: String, signature: String) in

        // Handle response
    })
}
```

```
//Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier:
String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier,
offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber,
keyIdentifier: String, signature: String) in

        // Handle response
    })
}
```

```
//Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier:
String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier,
offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber,
keyIdentifier: String, signature: String) in

        // Handle response
    })
}
```

```
//Generating the SKPaymentDiscount

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier:
String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier,
offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber,
keyIdentifier: String, signature: String) in

       let discountOffer = SKPaymentDiscount(identifier: offerIdentifier, keyIdentifier:
keyIdentifier, nonce: nonce, signature: signature, timestamp: timestamp)

        completion(discountOffer)
    })
}
```

```
//Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String,
withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
```

```
//Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String,
withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
```

```
//Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String,
withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
```

```
//Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String,
withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
```

```
//Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String,
withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
```

```
//Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String,
withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
```

```
//Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String,
withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
```

```
//Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String,
withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
```

# New SKError Codes

# New SKError Codes

| SKError.Code | Description |
| --- | --- |
| .invalidOfferIdentifier | Offer identifier cannot be found, or offer is inactive |

# New SKError Codes

| SKError.Code | Description |
| --- | --- |
| .invalidOfferIdentifier | Offer identifier cannot be found, or offer is inactive |
| .invalidOfferPrice | Pay-as-you-go offer price is higher than base subscription price |

# New SKError Codes

| SKError.Code | Description |
| --- | --- |
| .invalidOfferIdentifier | Offer identifier cannot be found, or offer is inactive |
| .invalidOfferPrice | Pay-as-you-go offer price is higher than base subscription price |
| .invalidSignature | Signature in SKPaymentDiscount did not successfully validate against public key |

# New SKError Codes

| SKError.Code | Description |
| --- | --- |
| .invalidOfferIdentifier | Offer identifier cannot be found, or offer is inactive |
| .invalidOfferPrice | Pay-as-you-go offer price is higher than base subscription price |
| .invalidSignature | Signature in SKPaymentDiscount did not successfully validate against public key |
| .missingOfferParams | One or more payload data items is missing (for example, empty applicationUsername) |

# References

Generating a Subscription Offer Using Node.js

Implementing Subscription Offers in Your App

Generating a Signature for Subscription Offers

---

In-App Purchases and Subscriptions Lab Friday, 3:00

# Determining Customer Eligibility

# Determining Eligibility

# Determining Eligibility

Must have previously subscribed

# Determining Eligibility

Must have previously subscribed

- Any auto-renewable subscription

# Determining Eligibility

Must have previously subscribed

- Any auto-renewable subscription
- Any subscription group

# Determining Eligibility

Must have previously subscribed

- Any auto-renewable subscription
- Any subscription group

Includes both current and lapsed subscribers

# Determining Eligibility

Must have previously subscribed

- Any auto-renewable subscription
- Any subscription group

Includes both current and lapsed subscribers

Eligible on first subscription period

# Determining Eligibility

Must have previously subscribed

- Any auto-renewable subscription
- Any subscription group

Includes both current and lapsed subscribers

Eligible on first subscription period

- Includes introductory offers

# Determining Eligibility

Must have previously subscribed

- Any auto-renewable subscription
- Any subscription group

Includes both current and lapsed subscribers

Eligible on first subscription period

- Includes introductory offers
- Includes free trials

```
//Creating the SKPaymentDiscount

{
    "status": 0,
    "receipt": {
        "in_app": [{
            "quantity": "1",
            "product_id": "com.example.apple.non_consumable_1",
        }, {
            "quantity": "1",
            "product_id": "com.example.apple.non_consumable_2",
        }, {
            "quantity": "1",
            "product_id": "com.example.apple.auto_renewable",
        }],
    }
}
```

```
//Creating the SKPaymentDiscount

{
    "status": 0,
    "receipt": {
        "in_app": [{
            "quantity": "1",
            "product_id": "com.example.apple.non_consumable_1",
        }, {
            "quantity": "1",
            "product_id": "com.example.apple.non_consumable_2",
        }, {
            "quantity": "1",
            "product_id": "com.example.apple.auto_renewable",
        }],
    }
}
```

```
//Creating the SKPaymentDiscount

{
    "status": 0,
    "receipt": {
        "in_app": [{
            "quantity": "1",
            "product_id": "com.example.apple.non_consumable_1",
        }, {
            "quantity": "1",
            "product_id": "com.example.apple.non_consumable_2",
        }, {
            "quantity": "1",
            "product_id": "com.example.apple.auto_renewable",
        }],
    }
}
```

```
//Creating the SKPaymentDiscount

{
    "status": 0,
    "receipt": {
        "in_app": [{
            "quantity": "1",
            "product_id": "com.example.apple.non_consumable_1",
        }, {
            "quantity": "1",
            "product_id": "com.example.apple.non_consumable_2",
        }, {
            "quantity": "1",
            "product_id": "com.example.apple.auto_renewable",
        }],
    }
}
```

# Eligibility, Distribution, and Reducing Churn Using Offers

Michael Gargas, Technical Advocate, App Store Commerce

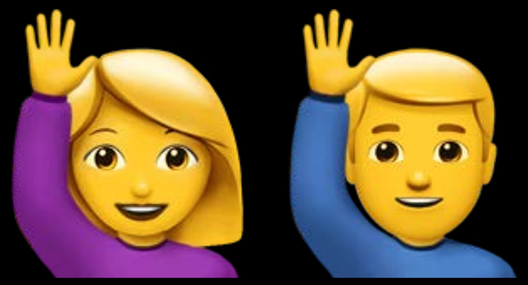
When do the majority of customers churn?

Should I offer an upgrade?

Have they subscribed in the past?

Have there been multiple billing issues?

How many consecutive renewal periods?



Is my customer engaged?

Has a customer churned?

Was a customer refunded?

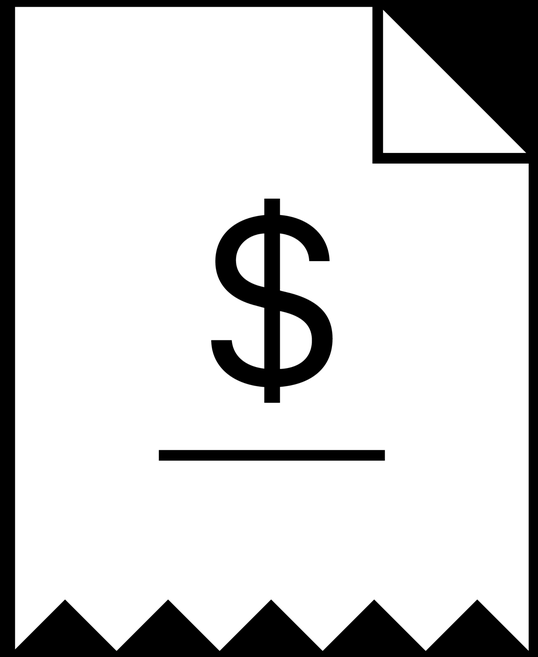
Is my price increase reasonable?

Why did a customer cancel?

Did the subscriber upgrade or downgrade?

# Determining Customer Eligibility

Receipt data

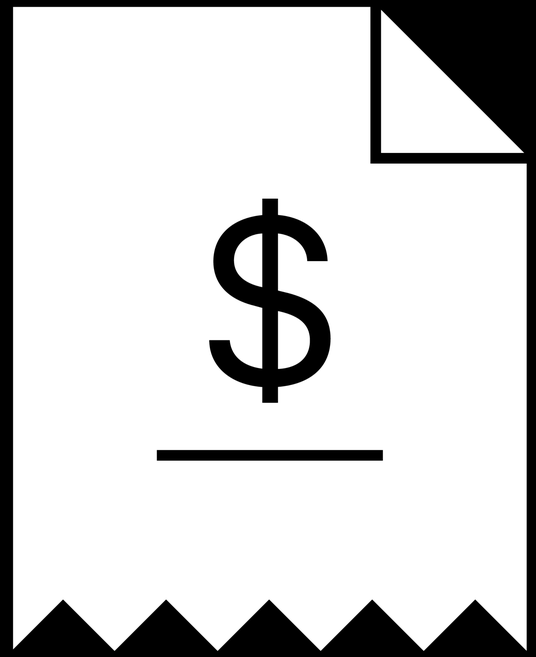


# Determining Customer Eligibility

## Receipt data

App Receipt

- in_app array
- latest_receipt_info



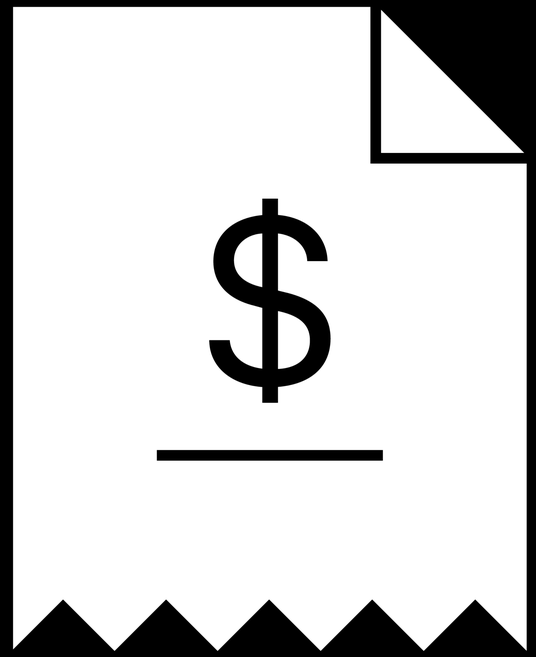
# Determining Customer Eligibility

## Receipt data

App Receipt

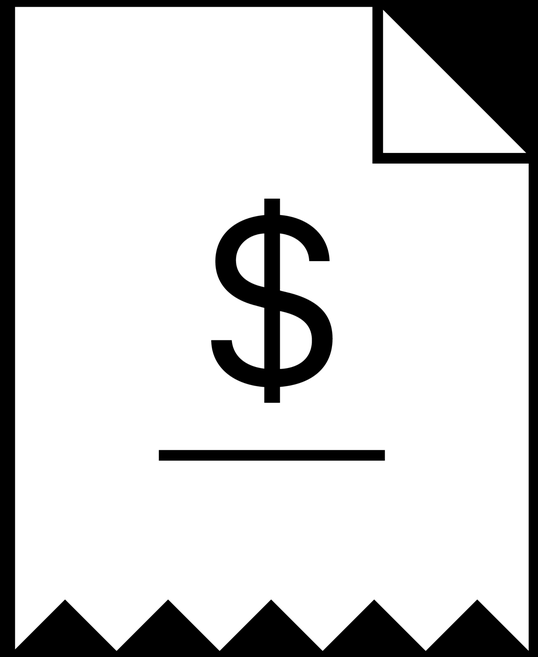
- in_app array
- latest_receipt_info

Server-to-server notifications



# Determining Customer Eligibility

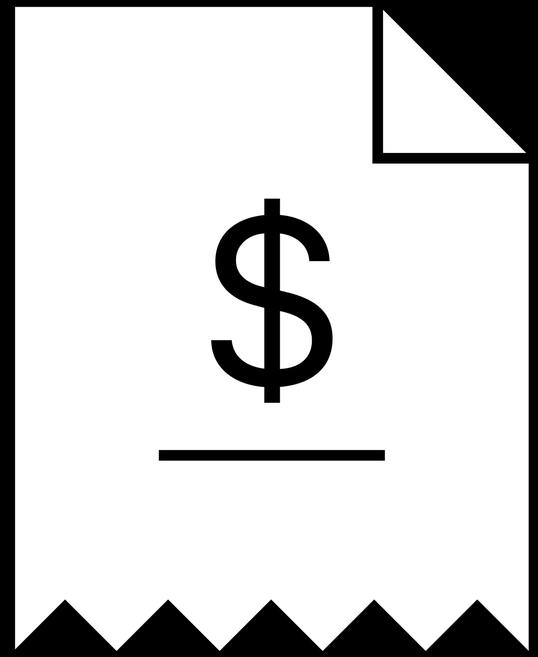
Receipt data



# Determining Customer Eligibility

Receipt data

Subscription status

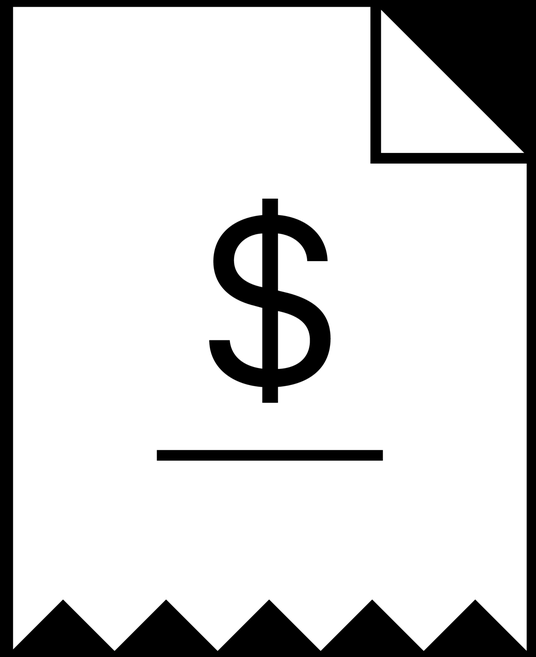


# Determining Customer Eligibility

Receipt data

Subscription status

Unique identifiers



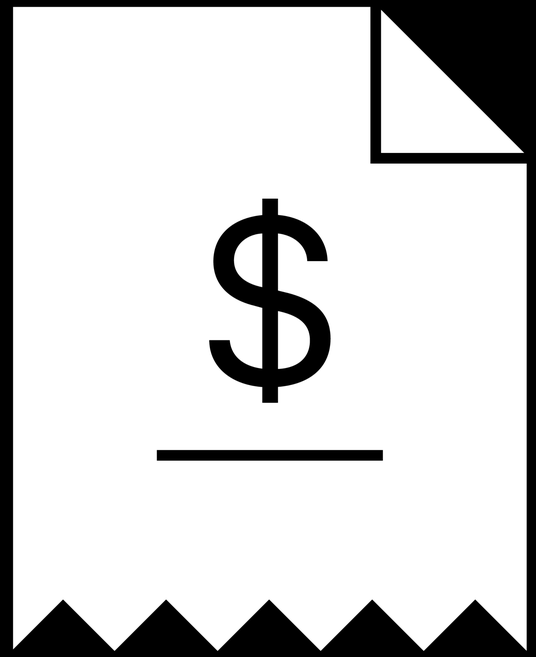
# Determining Customer Eligibility

Receipt data

Subscription status

Unique identifiers

Subscriber dates



# Determining Customer Eligibility

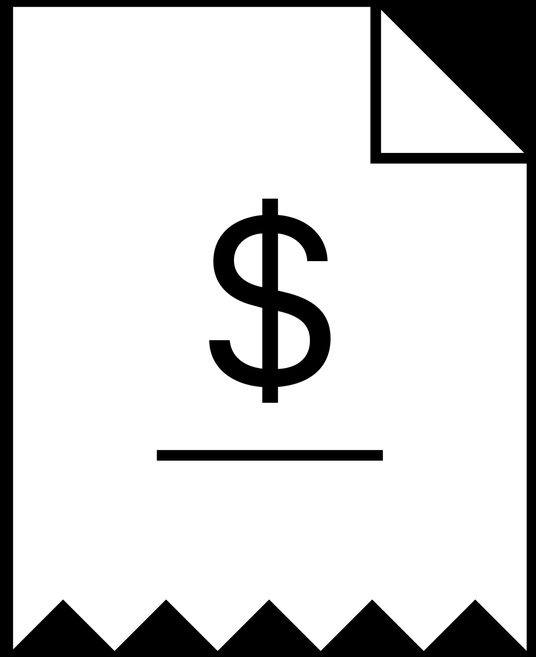
Receipt data

Subscription status

Unique identifiers

Subscriber dates

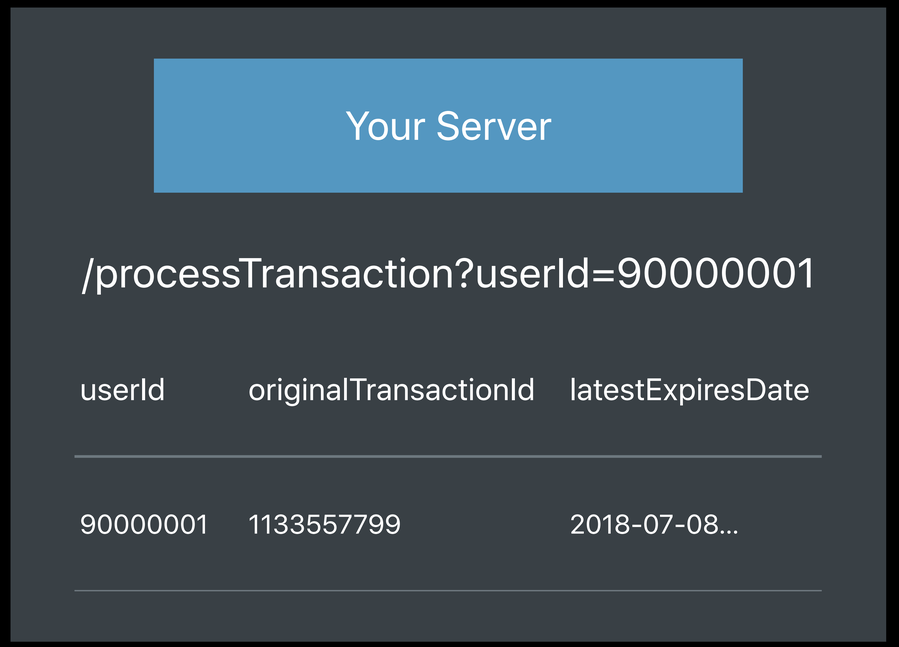
Subscriber intent



# Determining Customer Eligibility

## Storing user states

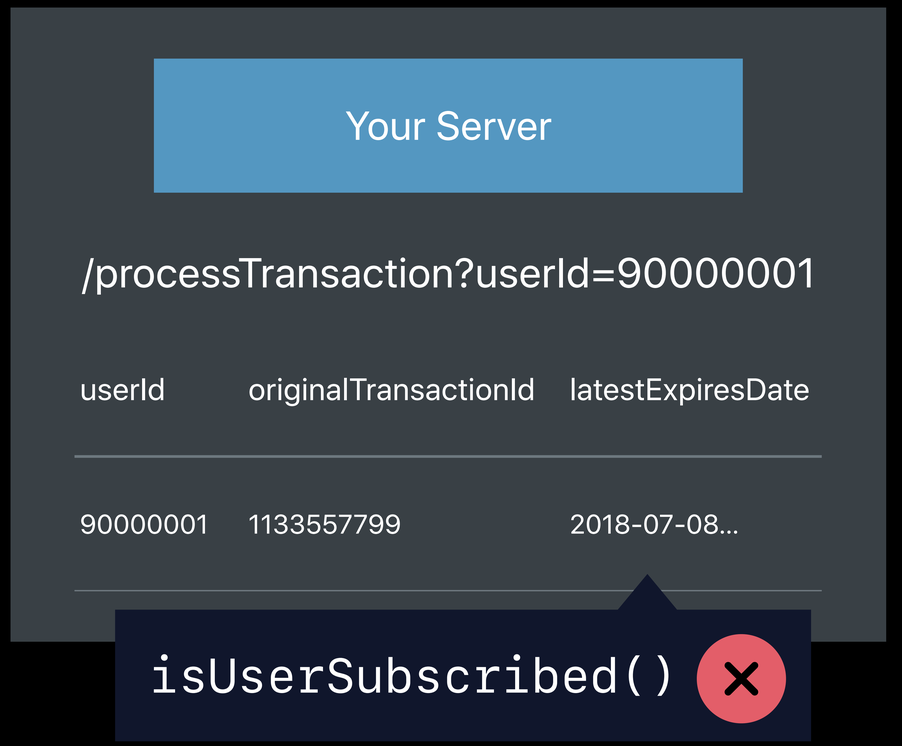
```
{ receipt: {
   in_app: [{
     transaction_id: “1234567890”,
     original_transaction_id: “1133557799”,
     expires_date: “2018-07-08…”, …
   }, {
     transaction_id: “2233445566”,
     original_transaction_id: “1133557799”,
     expires_date: “2018-08-08…”, …
   }]
  }
}
```



# Determining Customer Eligibility

## Storing user states

```
{ receipt: {
   in_app: [{
     transaction_id: “1234567890”,
     original_transaction_id: “1133557799”,
     expires_date: “2018-07-08…”, …
   }, {
     transaction_id: “2233445566”,
     original_transaction_id: “1133557799”,
     expires_date: “2018-08-08…”, …
   }]
  }
}
```



# Determining Customer Eligibility

Storing subscriber status fields

# Determining Customer Eligibility

Storing subscriber status fields

Save decoded JSON from `/verifyReceipt`

| userId | originalTransactionIdentifier | latestReceiptData | latestExpiresDate | latestDecodedReceipt |
|---|---|---|---|---|
| 90000001 | 1234567890 | d24Fs...kJ87dDGe3= | 2019-08-08... | { ... auto_renew_status: 1 ...} |

# Determining Customer Eligibility

Storing subscriber status fields

Save decoded JSON from `/verifyReceipt`

Parse out specific JSON fields

| userId | originalTransactionIdentifier | latestReceiptData | latestExpiresDate | autoRenewStatus |
|---|---|---|---|---|
| 90000001 | 1234567890 | d24Fs...kJ87dDGe3= | 2019-08-08... | 1 |

# Determining Customer Eligibility

Storing subscriber status fields

Save decoded JSON from `/verifyReceipt`

Parse out specific JSON fields

| userId | originalTransactionIdentifier | latestReceiptData | latestExpiresDate | autoRenewStatus |
|---|---|---|---|---|
| 90000001 | 1234567890 | d24Fs...kJ87dDGe3= | 2019-08-08... | 1 |

# Determining Customer Eligibility

Storing subscriber status fields

Save decoded JSON from `/verifyReceipt`

Parse out specific JSON fields

Save status changes from server-to-server notifications

| userId | originalTransactionIdentifier | latestReceiptData | latestExpiresDate | autoRenewStatus |
|---|---|---|---|---|
| 90000001 | 1234567890 | d24Fs...kJ87dDGe3= | 2019-08-08... | 1 |

# Determining Customer Eligibility

Developer data

# Determining Customer Eligibility

## Developer data

Developers have data Apple does not

Marry receipt data with customer subscription activity

Curate offers for specific cohorts of users

Monitor how users respond to offers using:

- Conversion
- Engagement
- Retention
- Churn

| userId | originalTransactionId | latestExpiresDate |
| --- | --- | --- |

| userId | originalTransactionId | latestExpiresDate |
| --- | --- | --- |
| 90000001 | 1133557799 | 2019-07-08... |
| 90000002 | 4820018625 | 2019-06-02... |
| 90000003 | 6090522426 | 2019-06-01... |
| 90000004 | 3092890314 | 2019-06-20... |
| 90000005 | 8426576390 | 2019-06-21... |
| 90000006 | 4286343643 | 2019-06-03... |

| userId | originalTransactionId | latestExpiresDate |
| --- | --- | --- |
| 90000001 | 1133557799 | 2019-07-08... |
| 90000002 | 4820018625 | 2019-06-02... |
| 90000003 | 6090522426 | 2019-06-01... |
| 90000004 | 3092890314 | 2019-06-20... |
| 90000005 | 8426576390 | 2019-06-21... |
| 90000006 | 4286343643 | 2019-06-03... |

| userId | originalTransactionId | latestExpiresDate | autoRenewStatus | billingRetryStatus |
|---|---|---|---|---|
| 90000001 | 1133557799 | 2019-07-08... | 0 | 0 |
| 90000002 | 4820018625 | 2019-06-02... | 1 | 0 |
| 90000003 | 6090522426 | 2019-06-01... | 0 | 0 |
| 90000004 | 3092890314 | 2019-06-20... | 0 | 0 |
| 90000005 | 8426576390 | 2019-06-21... | 1 | 1 |
| 90000006 | 4286343643 | 2019-06-03... | 0 | 0 |

| userId | originalTransactionId | latestExpiresDate | autoRenewStatus | billingRetryStatus | contentConsumed |
|---|---|---|---|---|---|
| 90000001 | 1133557799 | 2019-07-08... | 0 | 0 | 22.5 |
| 90000002 | 4820018625 | 2019-06-02... | 1 | 0 | 10 |
| 90000003 | 6090522426 | 2019-06-01... | 0 | 0 | 1 |
| 90000004 | 3092890314 | 2019-06-20... | 0 | 0 | 0 |
| 90000005 | 8426576390 | 2019-06-21... | 1 | 1 | 4 |
| 90000006 | 4286343643 | 2019-06-03... | 0 | 0 | 10 |

| userId | originalTransactionId | latestExpiresDate | autoRenewStatus | billingRetryStatus | contentConsumed |
|---|---|---|---|---|---|
| 90000001 | 1133557799 | 2019-07-08... | 0 | 0 | 22.5 |
| 90000002 | 4820018625 | 2019-06-02... | 1 | 0 | 10 |
| 90000003 | 6090522426 | 2019-06-01... | 0 | 0 | 1 |
| 90000004 | 3092890314 | 2019-06-20... | 0 | 0 | 0 |
| 90000005 | 8426576390 | 2019-06-21... | 1 | 1 | 4 |
| 90000006 | 4286343643 | 2019-06-03... | 0 | 0 | 10 |

| userId | originalTransactionId | latestExpiresDate | autoRenewStatus | billingRetryStatus | contentConsumed |
|---|---|---|---|---|---|
| 90000001 | 1133557799 | 2019-07-08... | 0 | 0 | 22.5 |
| 90000002 | 4820018625 | 2019-06-02... | 1 | 0 | |
| 90000003 | 6090522426 | 2019-06-01... | 0 | 0 | 1 |
| 90000004 | 3092890314 | 2019-06-20... | 0 | 0 | 0 |
| 90000005 | 8426576390 | 2019-06-21... | 1 | 1 | 4 |
| 90000006 | 4286343643 | 2019-06-03... | 0 | 0 | 10 |

isOffer1Eligible

isOffer3Eligible

# Determining Customer Eligibility

Customer cohorts

# Determining Customer Eligibility

## Customer cohorts

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | retentionOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|

# Determining Customer Eligibility

## Customer cohorts

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | retentionOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000001 | 0 | 0 | 0 | 0 | 5 | 1 | 0 | 0 |
| 90000002 | 6 | 0 | 0 | 0 | 125 | 0 | 1 | 0 |
| 90000003 | 10 | 1 | 0 | 2 | 352 | 0 | 0 | 1 |

# Determining Customer Eligibility

## Customer cohorts

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | retentionOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000001 | 0 | 0 | 0 | 0 | 5 | 1 | 0 | 0 |
| 90000002 | 6 | 0 | 0 | 0 | 125 | 0 | 1 | 0 |
| 90000003 | 10 | 1 | 0 | 2 | 352 | 0 | 0 | 1 |

# Determining Customer Eligibility

## Customer cohorts

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | retentionOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000001 | 0 | 0 | 0 | 0 | 5 | 1 | 0 | 0 |
| 90000002 | 6 | 0 | 0 | 0 | 125 | 0 | 1 | 0 |
| 90000003 | 10 | 1 | 0 | 2 | 352 | 0 | 0 | 1 |

# Determining Customer Eligibility

## Customer cohorts

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | retentionOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000001 | 0 | 0 | 0 | 0 | 5 | 1 | 0 | 0 |
| 90000002 | 6 | 0 | 0 | 0 | 125 | 0 | 1 | 0 |
| 90000003 | 10 | 1 | 0 | 2 | 352 | 0 | 0 | 1 |

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

Using offers to reduce churn


Identify
Engage
Fetch
Present
Purchase

# Distributing Offers

## Identify

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | retentionOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000001 | 0 | 0 | 0 | 0 | 5 | 1 | 0 | 0 |
| 90000002 | 6 | 0 | 0 | 0 | 125 | 0 | 1 | 0 |
| 90000003 | 10 | 1 | 0 | 2 | 352 | 0 | 0 | 1 |

# Distributing Offers

## Identify

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | retentionOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000001 | 0 | 0 | 0 | 0 | 5 | 1 | 0 | 0 |
| 90000002 | 6 | 0 | 0 | 0 | 125 | 0 | | |
| 90000003 | 10 | 1 | 0 | 2 | 352 | 0 | | |

isOfferEligible

isOfferEligible

isOfferEligible

# Distributing Offers

Presentation

In-App

- Messaging
- Notifications

# Distributing Offers

Presentation

# Distributing Offers

## Presentation

Landing point for offer consumption

Think about the call to action

Comply with App Review Guidelines for subscriptions


Identify
Circulate
Engage
Fetch
Present
Purchase


Identify
Circulate
Engage
Fetch
Present
Purchase


Identify
Circulate
Engage
Fetch
Present
Purchase

# Distributing Offers

Presenting

What's New in Universal Links

WWDC App

# Distributing Offers

## Presenting

External channels

- Paid advertising
- Email marketing

# Distributing Offers

## Presenting

External channels

- Paid advertising
- Email marketing

Universal links

- Drive users back into the app from external channels

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

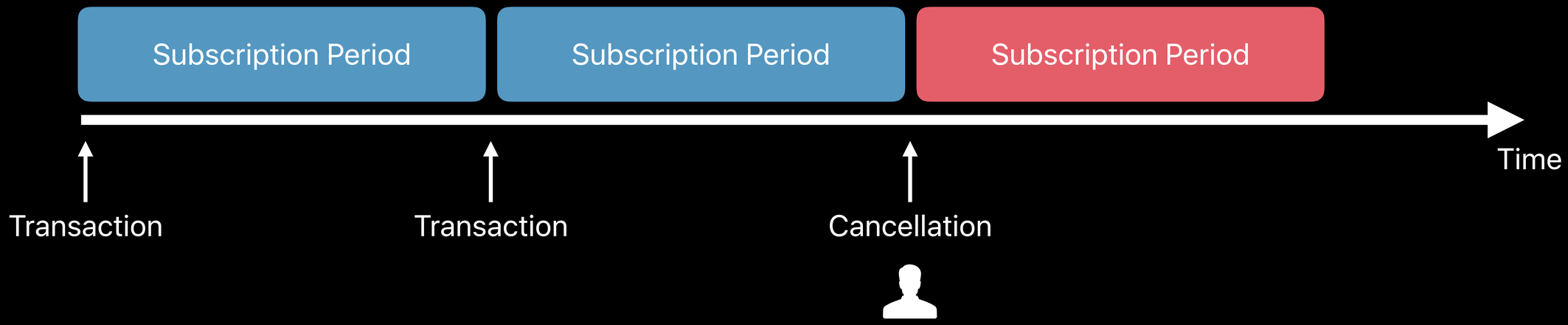
Using offers to reduce churn

# Using Offers to Reduce Churn

Voluntary churn

# Using Offers to Reduce Churn

## Voluntary churn



# Using Offers to Reduce Churn

Voluntary churn

# Using Offers to Reduce Churn

## Voluntary churn

Analyze customer subscription behavior

# Using Offers to Reduce Churn

## Voluntary churn

Analyze customer subscription behavior

Identify customer cohorts

# Using Offers to Reduce Churn

## Voluntary churn

Analyze customer subscription behavior

Identify customer cohorts

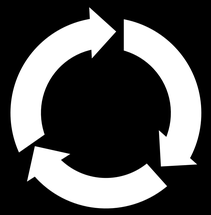
Distribute relevant subscription offers

# Using Offers to Reduce Churn

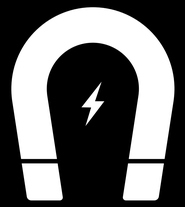
Offer strategies

# Using Offers to Reduce Churn

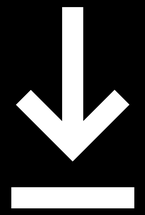
## Offer strategies



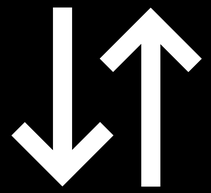
Win Backs



Retention Marketing



Save Offers



Upgrade/Downgrade

Customer Service

Loyalty

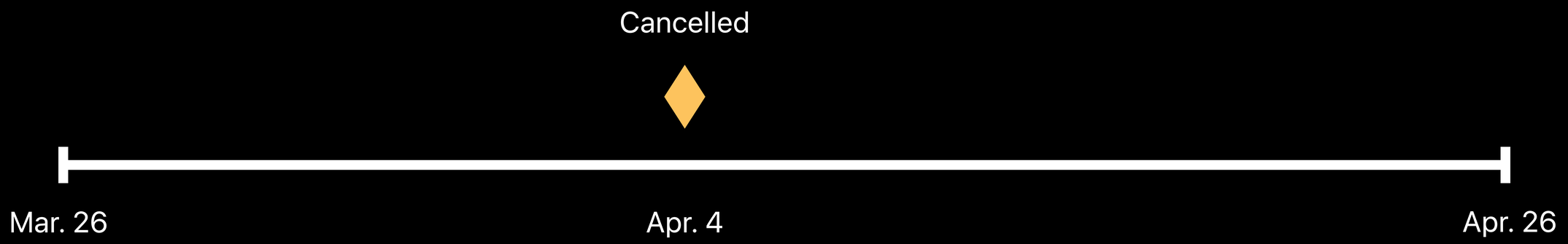
# Using Offers to Reduce Churn

## Winback


```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”,
}]
```

# Using Offers to Reduce Churn

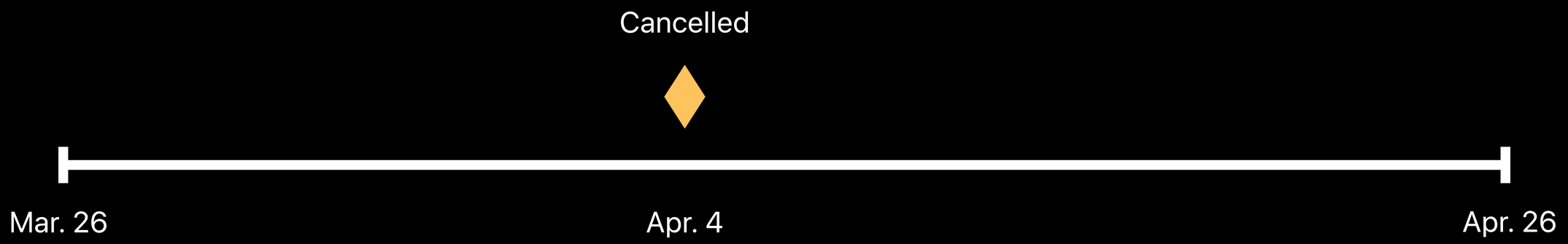
## Winback




```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”,
  “cancellation_date”: “2019-04-04…”,
  “cancellation_reason”: “1”, …
}]
```

# Using Offers to Reduce Churn

## Winback

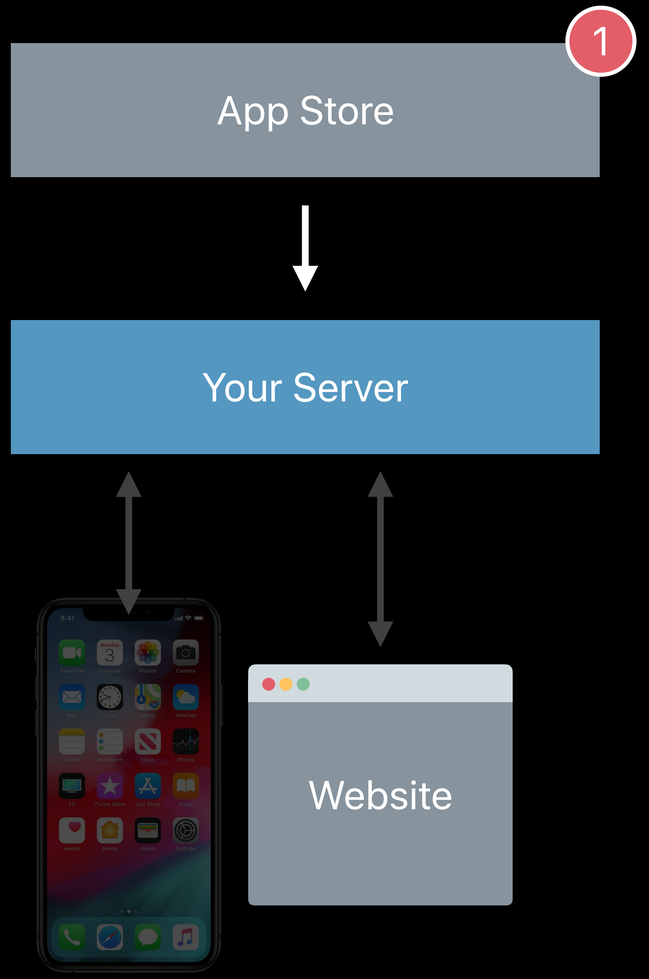



```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”,
  “cancellation_date”: “2019-04-04…”,
  “cancellation_reason”: “1”, …
}]
```

# Using Offers to Reduce Churn

## Server-to-server notification

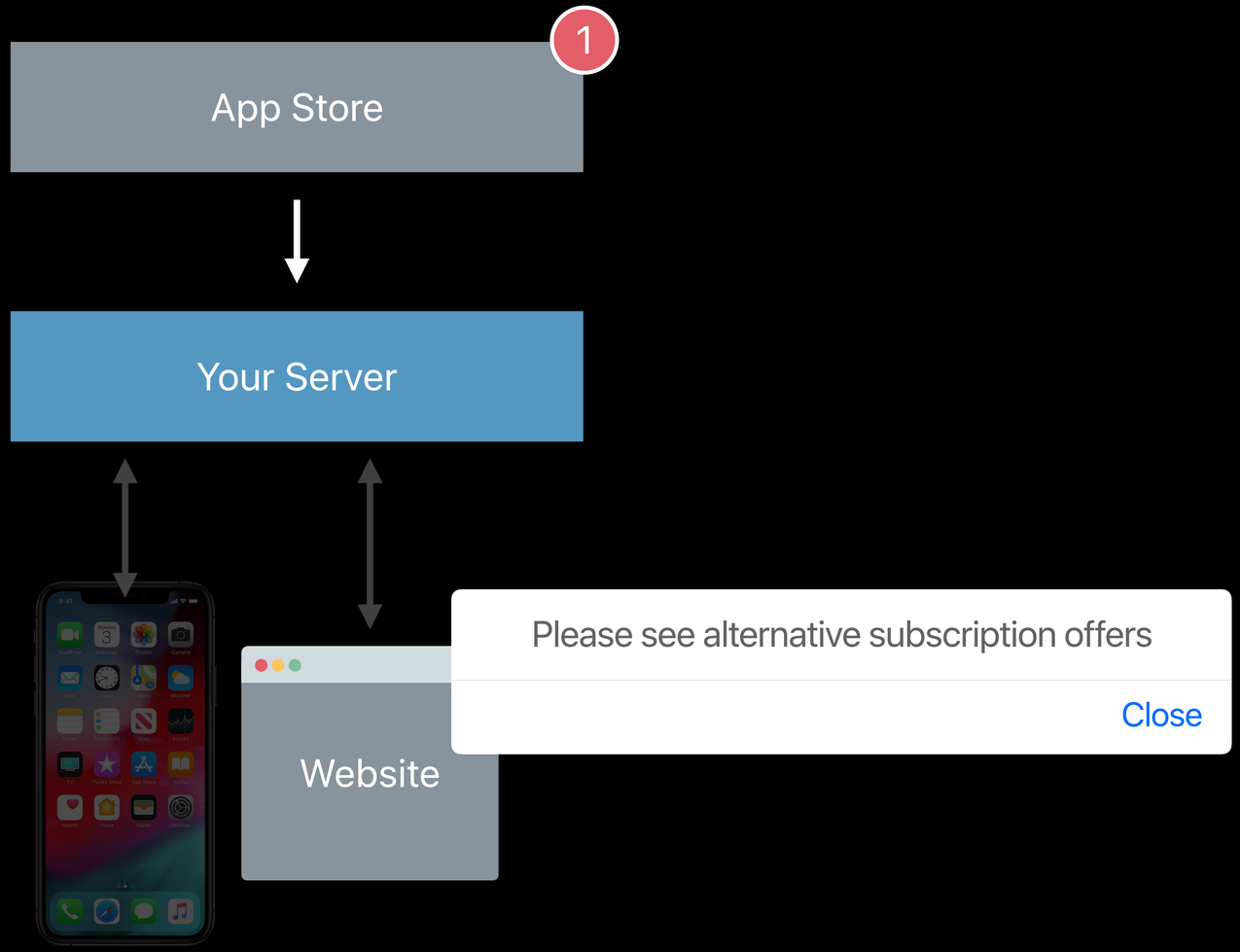
```
{ notification_type: “CANCEL”, … }
```



# Using Offers to Reduce Churn

## Server-to-server notification

```
{ notification_type: “CANCEL”, … }
```



# Using Offers to Reduce Churn

## Winback

```
pending_renewal_info: [{
  “original_transaction_id”: “1133557799”,
  “product_id”: “plus_subscription_999_6m”,
  “auto_renew_status”: “0”
  “cancellation_date”: “2019-04-04…”,
  “cancellation_reason”: “1”,
  “expiration_intent”: “1”, …
}]
```

Your Server

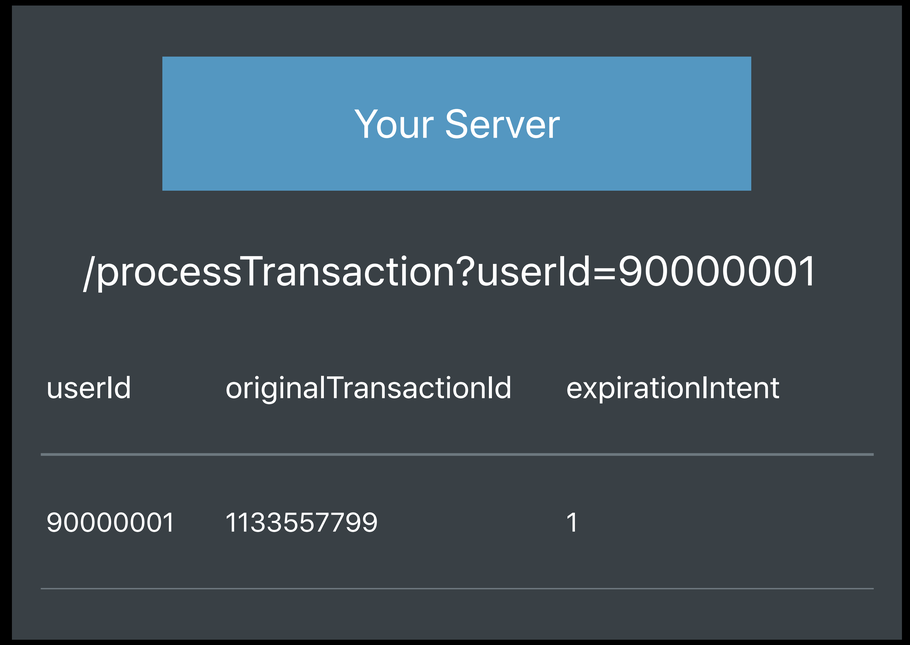
/processTransaction?userId=90000001

| userId | originalTransactionId | expirationIntent |
| --- | --- | --- |
| 90000001 | 1133557799 | 1 |

# Using Offers to Reduce Churn

## Winback

```
pending_renewal_info: [{
  “original_transaction_id”: “1133557799”,
  “product_id”: “plus_subscription_999_6m”,
  “auto_renew_status”: “0”
  “cancellation_date”: “2019-04-04…”,
  “cancellation_reason”: “1”,
  “expiration_intent”: “1”, …
}]
```



# Using Offers to Reduce Churn

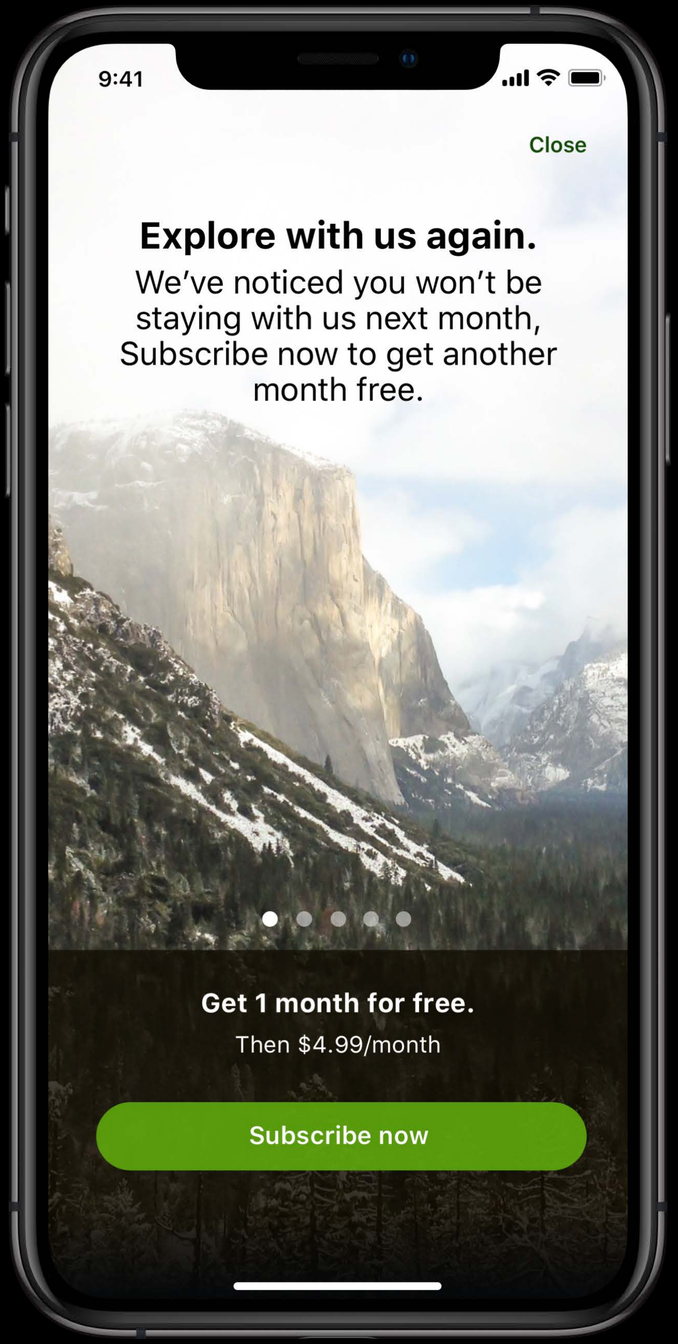
## Winback

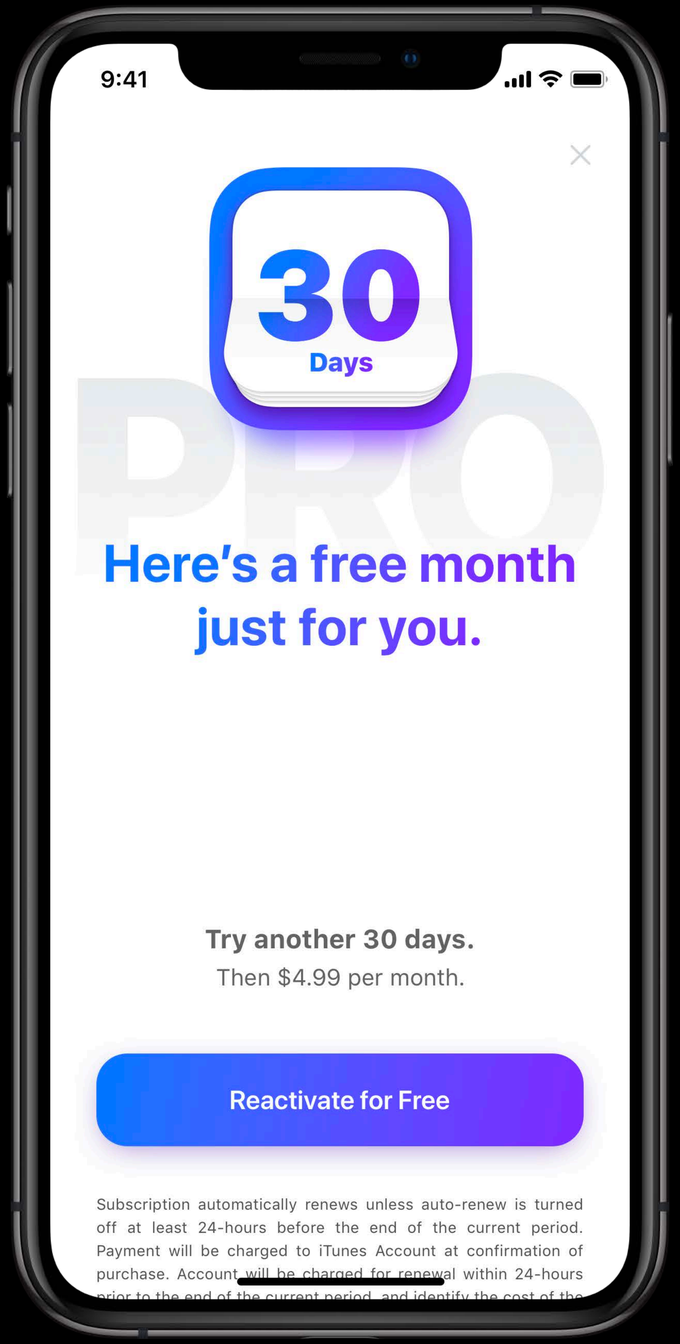

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000003 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 |

# Using Offers to Reduce Churn

## Winback


| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000003 | 0 | 0 | 0 | 0 | 4 | 1 | 0 | 0 |


9:41
Close
Explore with us again.
We've noticed you won't be staying with us next month, Subscribe now to get another month free.
Get 1 month for free.
Then $4.99/month
Subscribe now


9:41
30
Days
PRO
Here's a free month just for you.
Try another 30 days.
Then $4.99 per month.
Reactivate for Free
Subscription automatically renews unless auto-renew is turned off at least 24-hours before the end of the current period. Payment will be charged to iTunes Account at confirmation of purchase. Account will be charged for renewal within 24-hours

# Using Offers to Reduce Churn

## Retention


```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”,…
}]
```

# Using Offers to Reduce Churn

## Retention


```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”,…
}]
```

# Using Offers to Reduce Churn

## Retention


# Using Offers to Reduce Churn

## Retention


# Using Offers to Reduce Churn

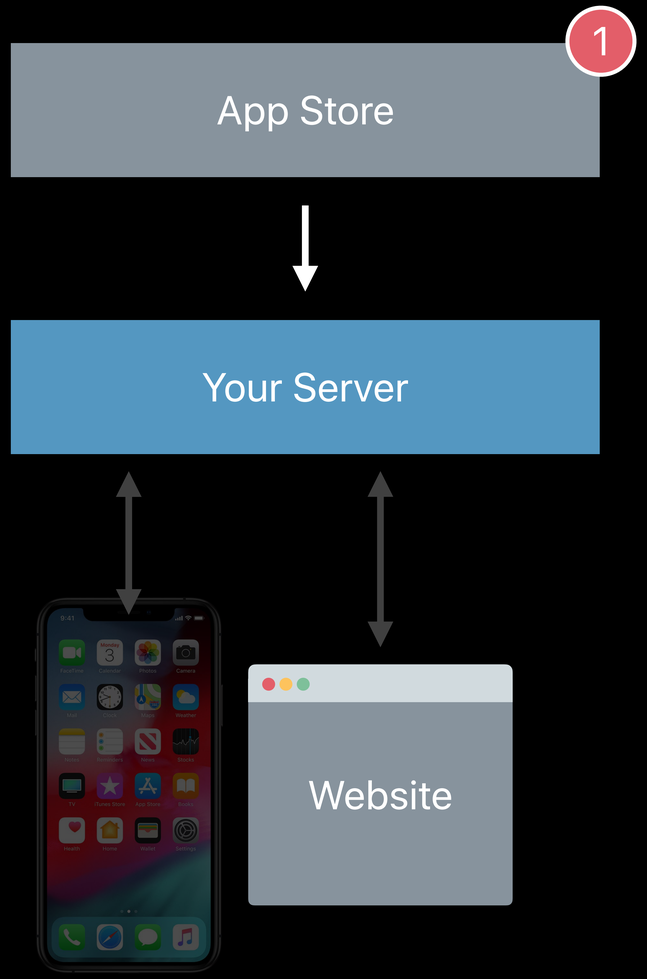
## Retention


```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”
  “auto_renew_status”: "0",…
}]
```

# Using Offers to Reduce Churn

## Server-to-server notification

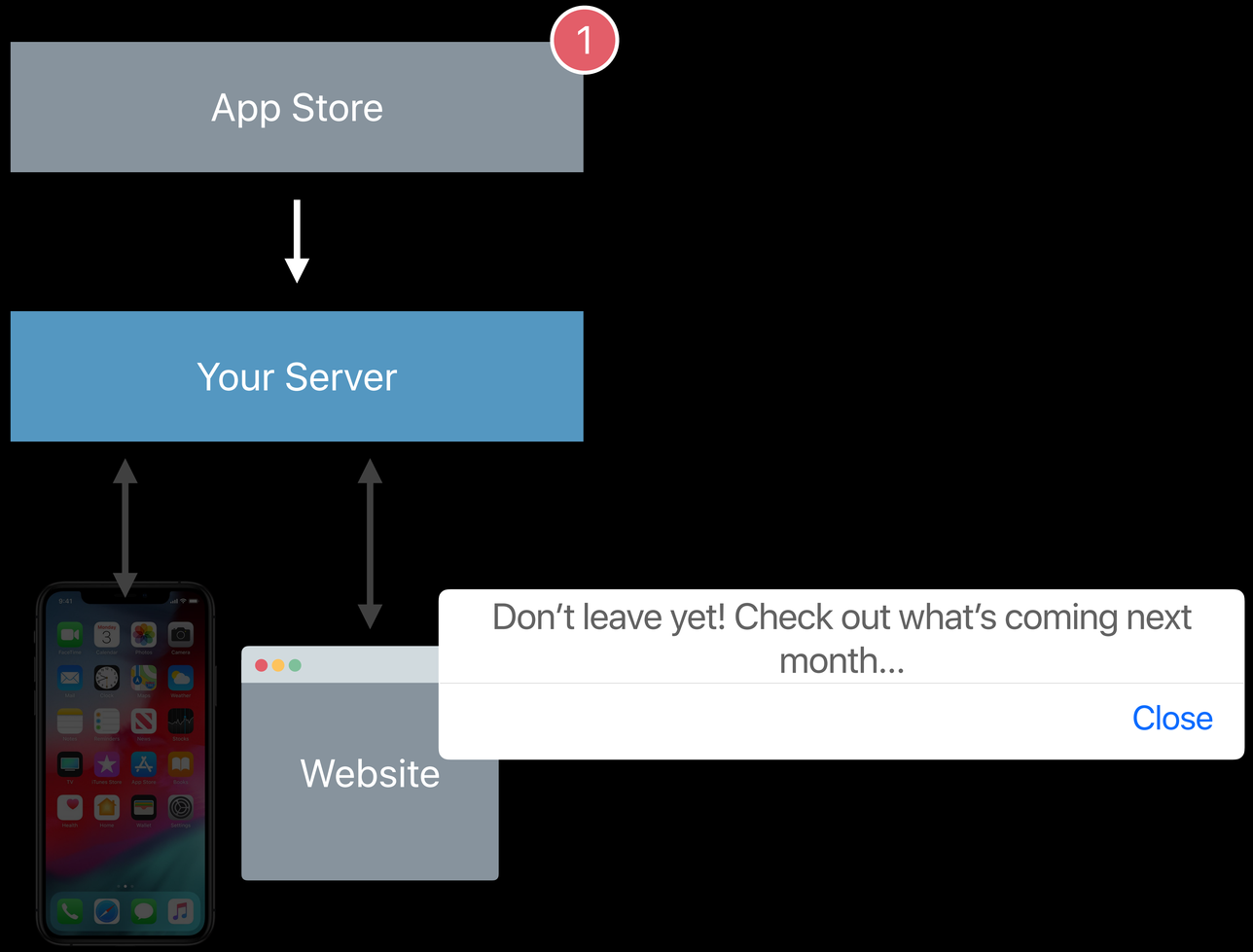
```
{ notification_type:
“DID_CHANGE_RENEWAL_STATUS”, … }
```



# Using Offers to Reduce Churn

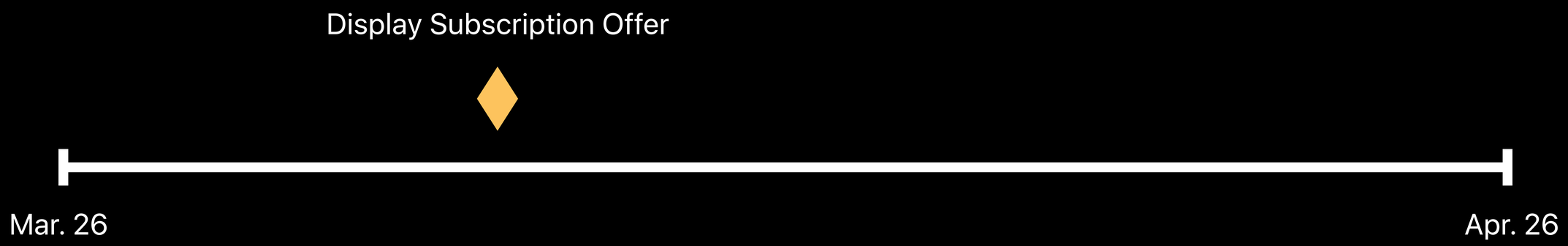
## Server-to-server notification

```
{ notification_type:
“DID_CHANGE_RENEWAL_STATUS”, … }
```



# Using Offers to Reduce Churn

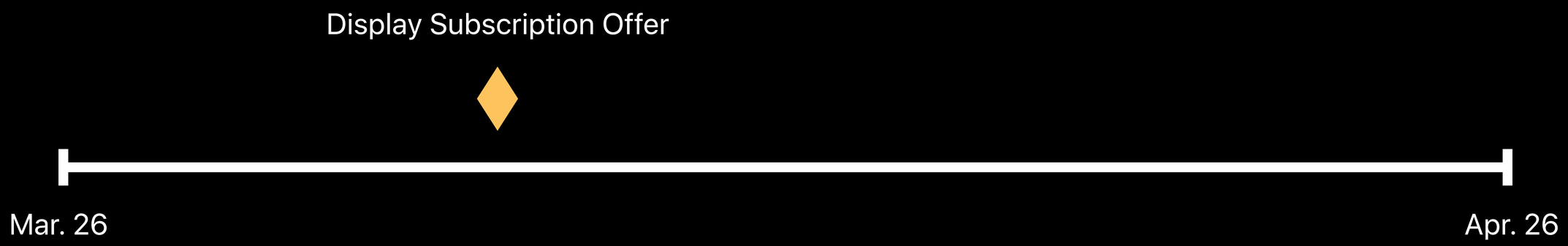
## Retention



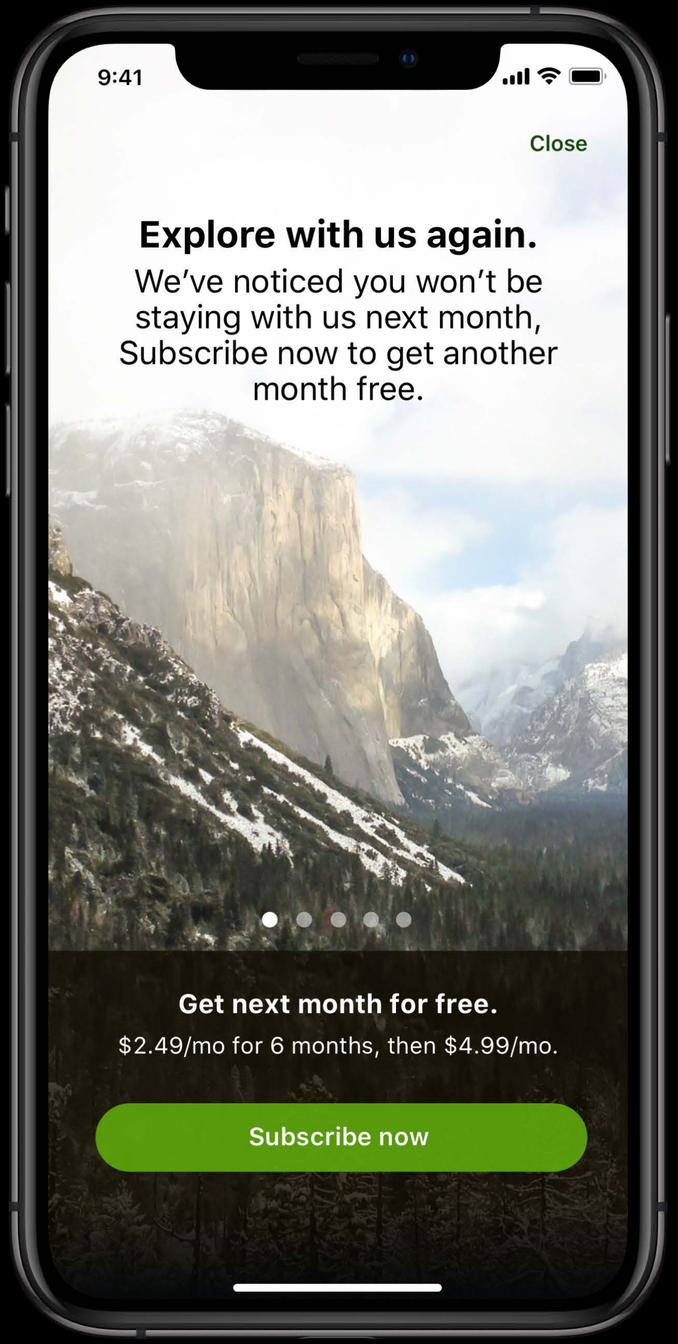

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | retentionOffer |
|---|---|---|---|---|---|---|---|---|
| 90000003 | 4 | 1 | 0 | 0 | 56 | 0 | 0 | 0 |

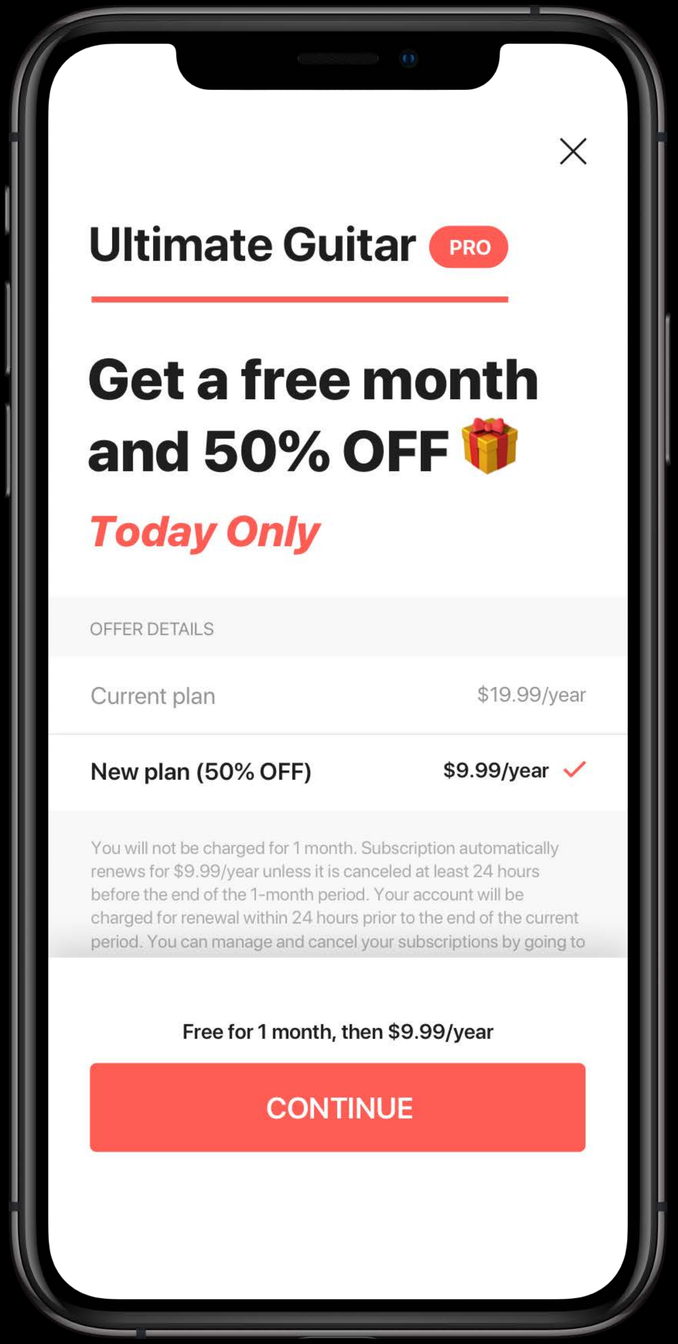
# Using Offers to Reduce Churn

## Retention




| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | retentionOffer |
|---|---|---|---|---|---|---|---|---|
| 90000003 | 4 | 0 | 0 | 0 | 56 | 0 | 0 | 1 |


9:41
Close
Explore with us again.
We’ve noticed you won’t be staying with us next month, Subscribe now to get another month free.
Get next month for free.
$2.49/mo for 6 months, then $4.99/mo.
Subscribe now


Ultimate Guitar
PRO
Get a free month and 50% OFF
Today Only
OFFER DETAILS
Current plan
$19.99/year
New plan (50% OFF)
$9.99/year
You will not be charged for 1 month. Subscription automatically renews for $9.99/year unless it is canceled at least 24 hours before the end of the 1-month period. Your account will be charged for renewal within 24 hours prior to the end of the current period. You can manage and cancel your subscriptions by going to
Free for 1 month, then $9.99/year
CONTINUE

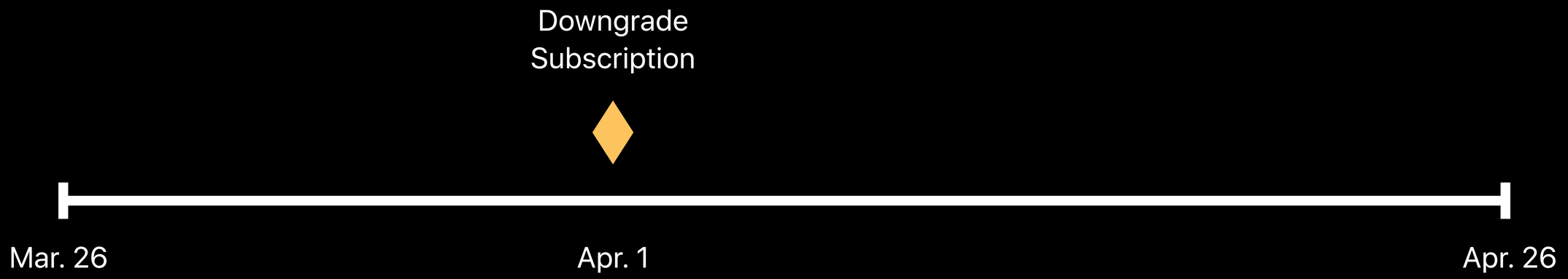
# Using Offers to Reduce Churn

## Upgrade


```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”,…
}]
```

# Using Offers to Reduce Churn

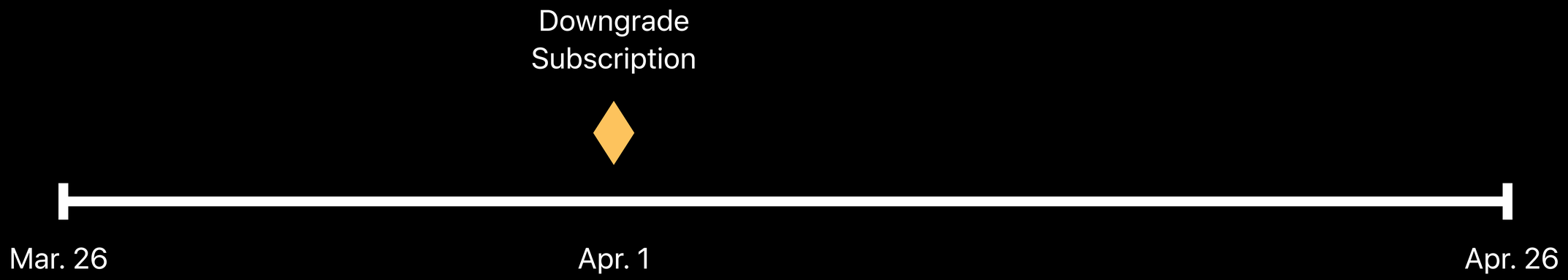
## Upgrade




```
pending_renewal_info: [{
  “original_transaction_id”: “1133557799”,
  “product_id”: “plus_subscription_999_6m",
  “auto_renew_status”: “1”,
  “auto_renew_product_id”: “com.your.product.id”, …
}]
```

# Using Offers to Reduce Churn

## Upgrade




```
pending_renewal_info: [{
  “original_transaction_id”: “1133557799”,
  “product_id”: “plus_subscription_999_6m",
  “auto_renew_status”: “1”,
  “auto_renew_product_id”: “com.your.product.id”, …
}]
```

# Auto Renew Preference

```
pending_renewal_info: [{
  “original_transaction_id”: “1133557799”,
  “product_id”: “plus_subscription_999_6m",
  “auto_renew_status”: “1”,
  “auto_renew_product_id”: “com.your.product.id“, …
}]
```

Your Server

/processTransaction?userId=90000001

| userId | originalTransactionId | autoRenewProductID |
| --- | --- | --- |
| 90000001 | 1133557799 | com.your.product.id |

# Auto Renew Preference

```
pending_renewal_info: [{
  “original_transaction_id”: “1133557799”,
  “product_id”: “plus_subscription_999_6m",
  “auto_renew_status”: “1”,
  “auto_renew_product_id”: “com.your.product.id“, …
}]
```

Your Server

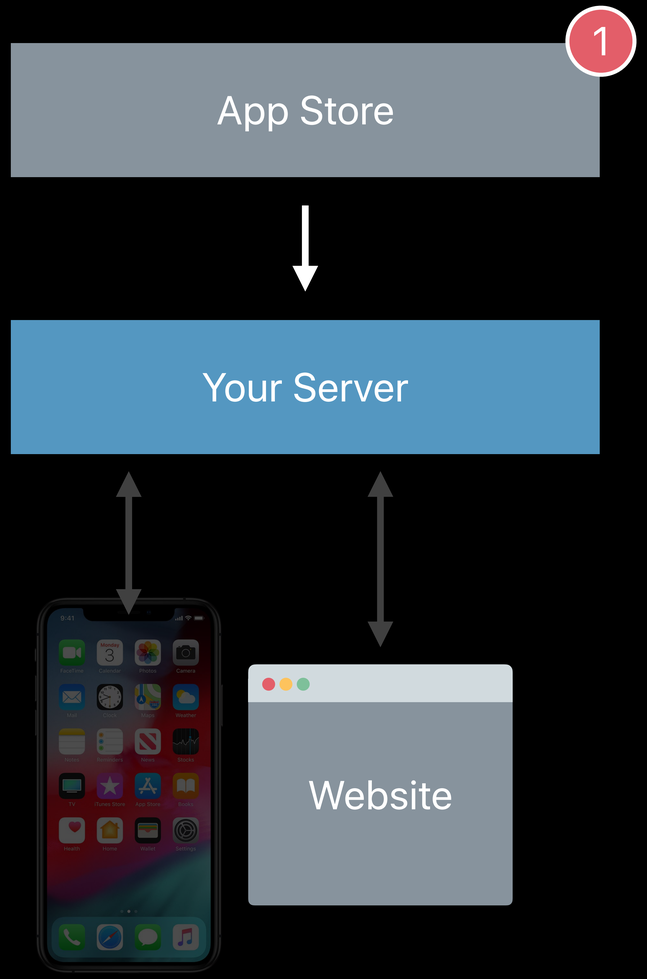
/processTransaction?userId=90000001

| userId | originalTransactionId | autoRenewProductID |
| --- | --- | --- |
| 90000001 | 1133557799 | com.your.product.id |

# Using Offers to Reduce Churn

## Server-to-server notification

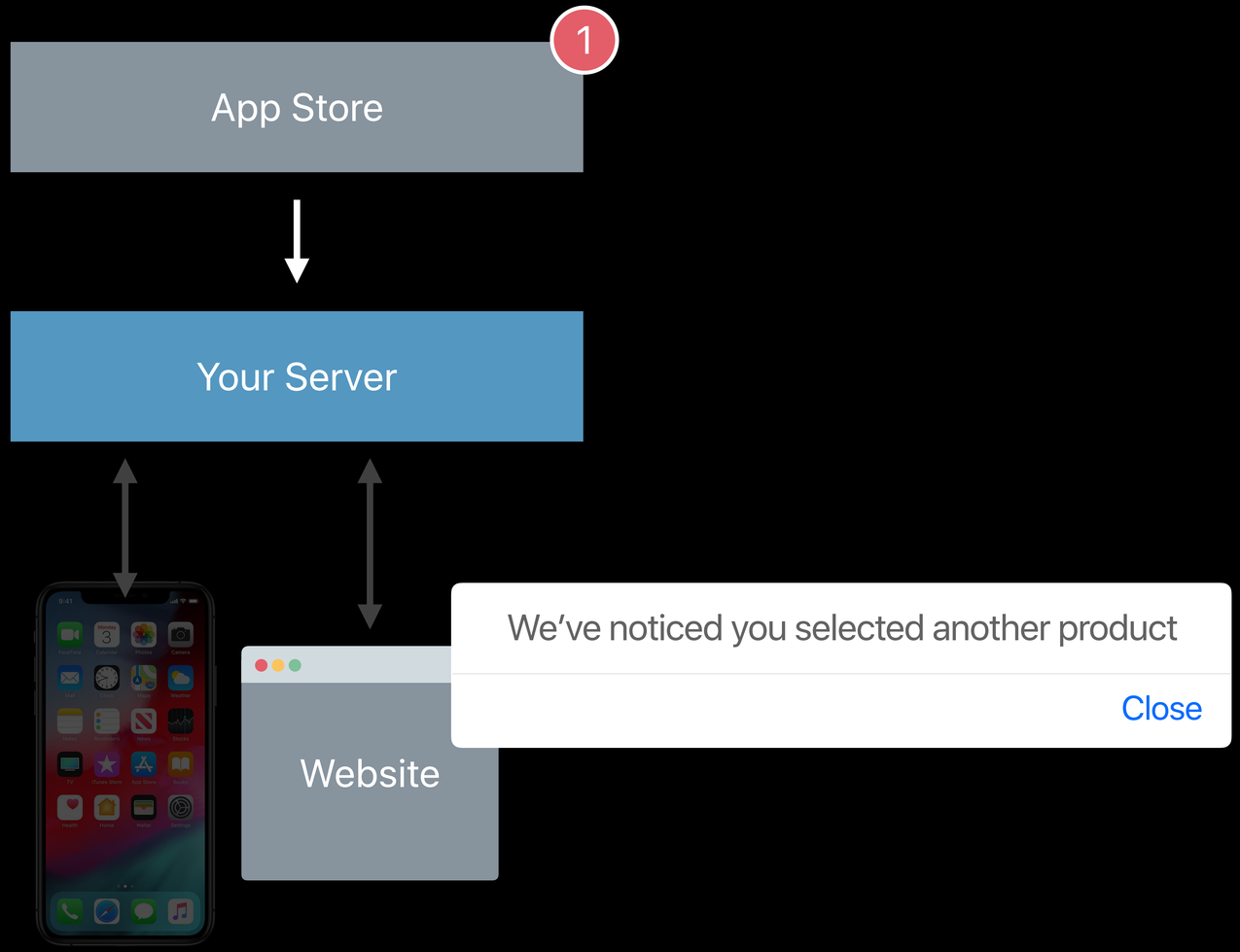
```
{ notification_type:
“DID_CHANGE_RENEWAL_PREFERENCE”, … }
```



# Using Offers to Reduce Churn

## Server-to-server notification

```
{ notification_type:
“DID_CHANGE_RENEWAL_PREFERENCE”, … }
```

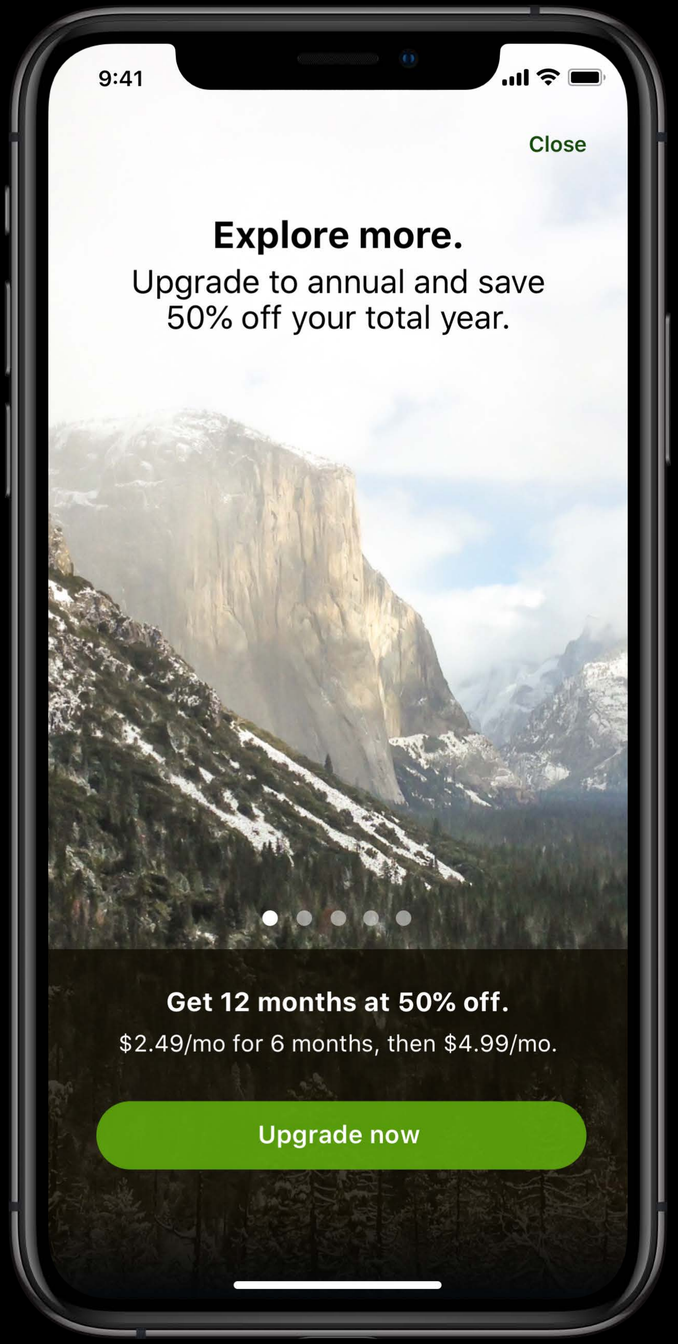


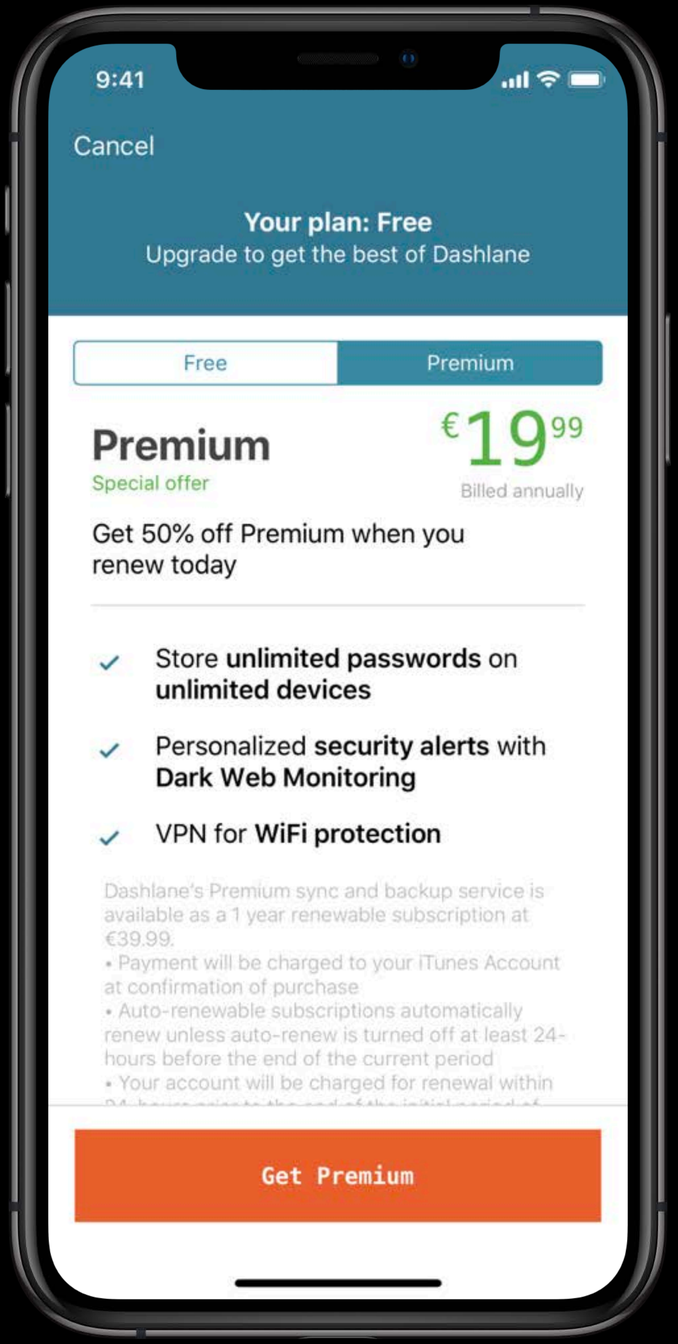
# Example Subscription


| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000003 | 11 | 1 | 0 | 2 | 352 | 0 | 0 | 1 |

# Example Subscription


| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000003 | 11 | 1 | 0 | 2 | 352 | 0 | 0 | 1 |


9:41
Close
Explore more.
Upgrade to annual and save
50% off your total year.
Get 12 months at 50% off.
$2.49/mo for 6 months, then $4.99/mo.
Upgrade now


9:41
Cancel
Your plan: Free
Upgrade to get the best of Dashlane
Free
Premium
Premium
Special offer
€19.99
Billed annually
Get 50% off Premium when you renew today
Store unlimited passwords on unlimited devices
Personalized security alerts with Dark Web Monitoring
VPN for WiFi protection
Dashlane's Premium sync and backup service is available as a 1 year renewable subscription at €39.99.
• Payment will be charged to your iTunes Account at confirmation of purchase
• Auto-renewable subscriptions automatically renew unless auto-renew is turned off at least 24-hours before the end of the current period
• Your account will be charged for renewal within
Get Premium

# Using Offers to Reduce Churn

Customer service

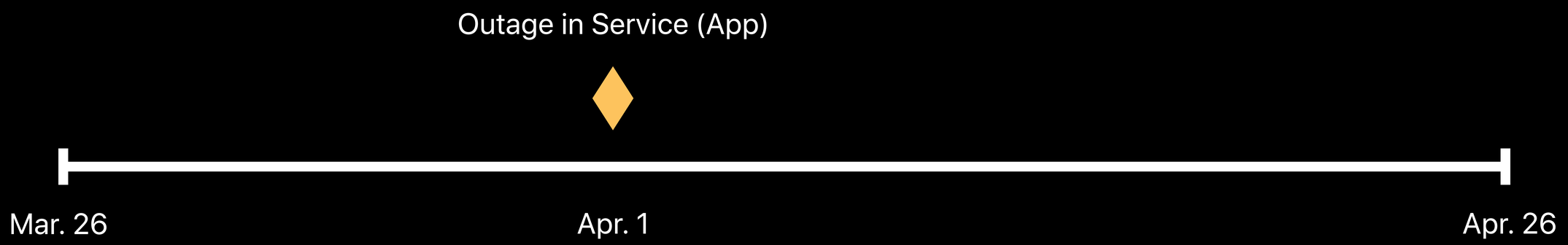
# Using Offers to Reduce Churn

## Customer service


```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”,…
}]
```

# Using Offers to Reduce Churn

## Customer service




```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”,…
}]
```

| userId | originalTransactionId | latestExpiresDate | autoRenewStatus | billingRetryStatus | contentConsumed |
|---|---|---|---|---|---|
| 90000001 | 1133557799 | 2019-07-08... | 0 | 0 | 22.5 |
| 90000002 | 4820018625 | 2019-06-02... | 1 | 0 | 10 |
| 90000003 | 6090522426 | 2019-06-01... | 0 | 0 | 1 |
| 90000004 | 3092890314 | 2019-06-20... | 0 | 0 | 0 |
| 90000005 | 8426576390 | 2019-06-21... | 1 | 1 | 4 |
| 90000006 | 4286343643 | 2019-06-03... | 0 | 0 | 10 |

| userId | originalTransactionId | latestExpiresDate | autoRenewStatus | billingRetryStatus | contentConsumed |
|---|---|---|---|---|---|
| 90000001 | 1133557799 | 2019-07-08... | 0 | 0 | 22.5 |
| 90000002 | 4820018625 | 2019-06-02... | 1 | 0 | 10 |
| 90000003 | 6090522426 | 2019-06-01... | 0 | 0 | 1 |
| 90000004 | 3092890314 | 2019-06-20... | 0 | 0 | 0 |
| 90000005 | 8426576390 | 2019-06-21... | 1 | 1 | 4 |
| 90000006 | 4286343643 | 2019-06-03... | 0 | 0 | |

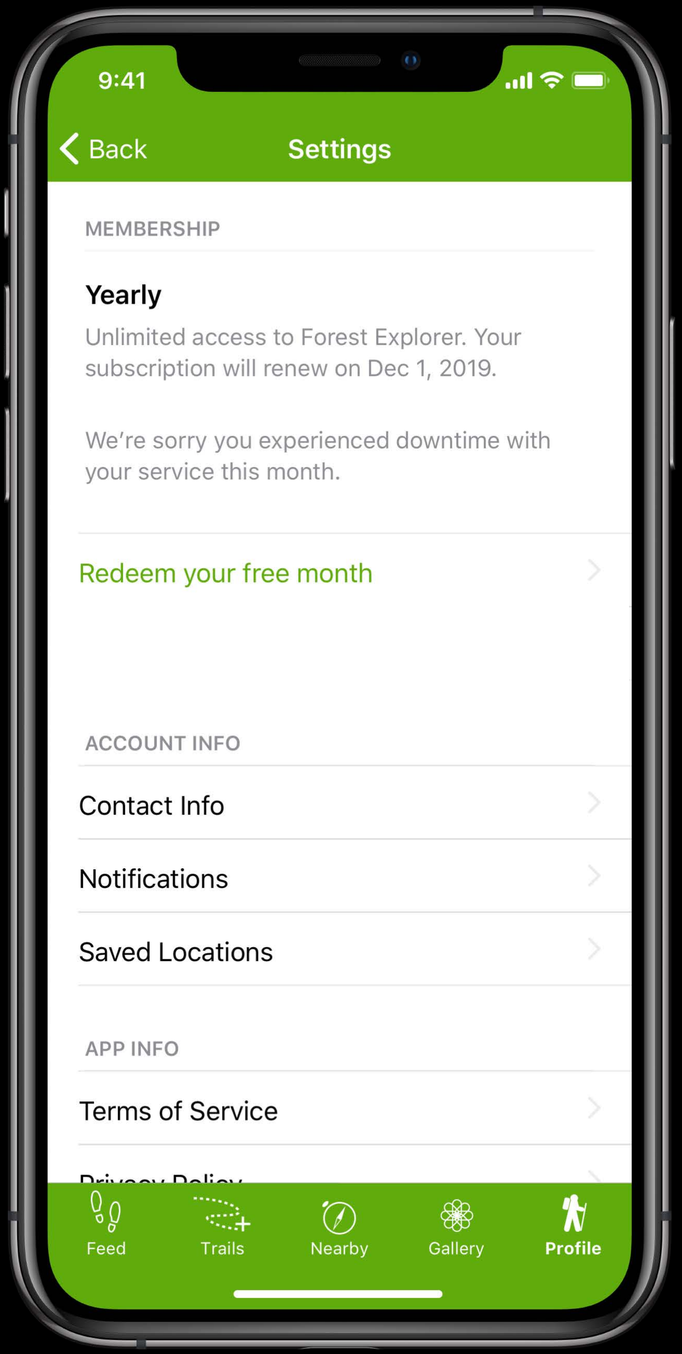
isOffer3Eligible

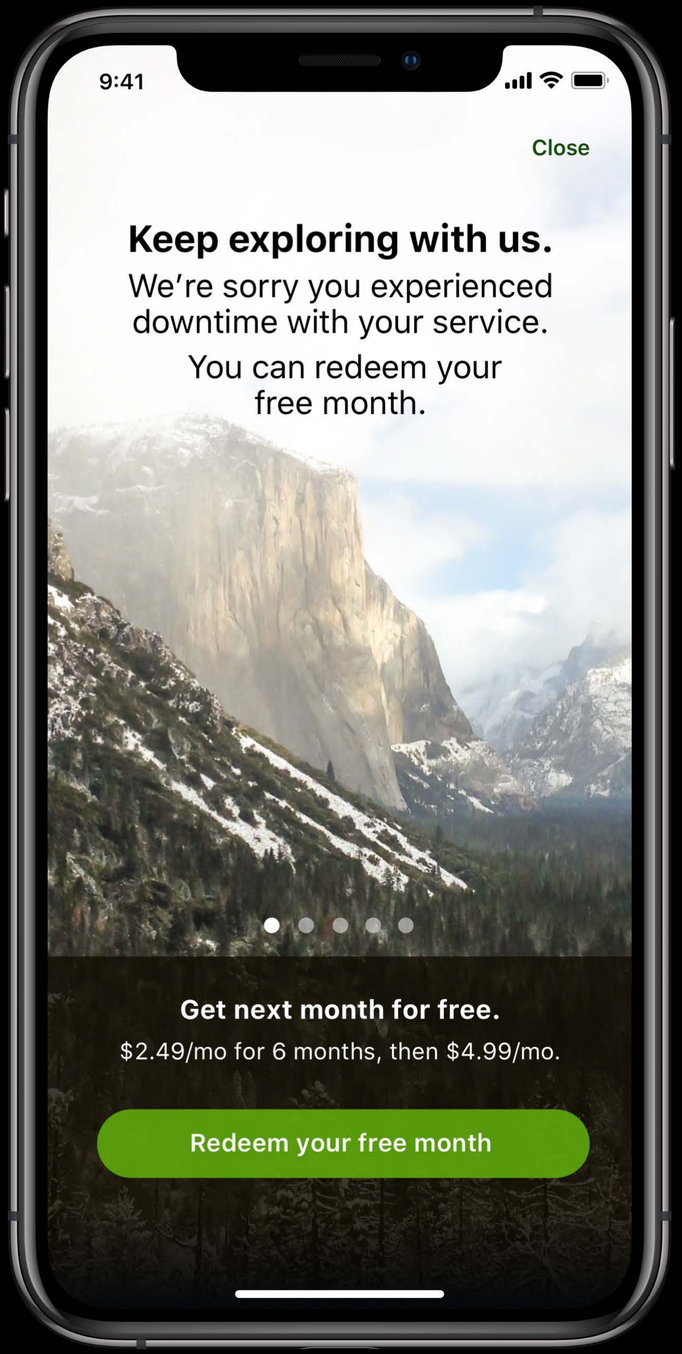
# Example Subscription


| userId | originalTransactionId | latestExpiresDate | autoRenewStatus | billingRetryStatus | hoursViewed | offerEligible |
|---|---|---|---|---|---|---|
| 90000005 | 8426576390 | 2019-06-21... | 1 | 1 | 4 | 1 |

# Example Subscription


| userId | originalTransactionId | latestExpiresDate | autoRenewStatus | billingRetryStatus | hoursViewed | offerEligible |
|---|---|---|---|---|---|---|
| 90000005 | 8426576390 | 2019-06-21... | 1 | 1 | 4 | 1 |


9:41
Back
Settings
MEMBERSHIP
Yearly
Unlimited access to Forest Explorer. Your subscription will renew on Dec 1, 2019.
We're sorry you experienced downtime with your service this month.
Redeem your free month
ACCOUNT INFO
Contact Info
Notifications
Saved Locations
APP INFO
Terms of Service
Feed
Trails
Nearby
Gallery
Profile


9:41
Close
Keep exploring with us.
We're sorry you experienced downtime with your service.
You can redeem your free month.
Get next month for free.
$2.49/mo for 6 months, then $4.99/mo.
Redeem your free month

# Using Offers to Reduce Churn

## Loyalty


```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”,…
}]
```

# Using Offers to Reduce Churn

## Loyalty


```
pending_renewal_info: [{
  “original_transaction_id”: “1133557799”,
  “product_id”: “plus_subscription_999_6m",
  “auto_renew_status”: “1”,
}]
```

# Using Offers to Reduce Churn

## Storing consecutive renewals

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000001 | 0 | 0 | 0 | 0 | 5 | 1 | 0 | 0 |
| 90000002 | 6 | 0 | 0 | 0 | 125 | 0 | 0 | 1 |
| 90000003 | 11 | 1 | 0 | 2 | 352 | 0 | 1 | 0 |

# Using Offers to Reduce Churn

## Storing consecutive renewals

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000001 | 0 | 0 | 0 | 0 | 5 | 1 | 0 | 0 |
| 90000002 | 6 | 0 | 0 | 0 | 125 | 0 | 0 | 1 |
| 90000003 | 11 | 1 | 0 | 2 | 352 | 0 | 1 | 0 |

# Using Offers to Reduce Churn

## Storing consecutive renewals

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000001 | 0 | 0 | 0 | 0 | 5 | 1 | 0 | 0 |
| 90000002 | 6 | 0 | 0 | 0 | 125 | 0 | 0 | 1 |
| 90000003 | 11 | 1 | 0 | 2 | 352 | 0 | 1 | 0 |

# Using Offers to Reduce Churn

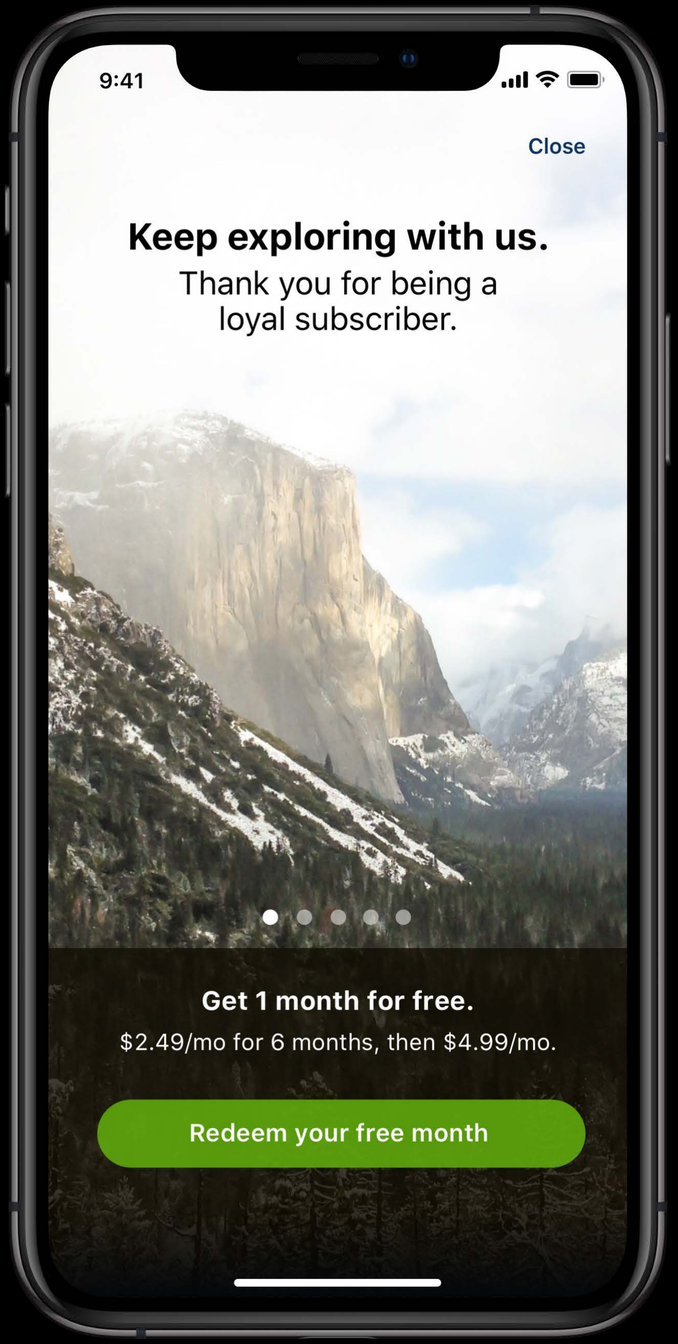
## Loyalty


| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000003 | 11 | 1 | 0 | 2 | 352 | 0 | 1 | 0 |

# Using Offers to Reduce Churn

## Loyalty

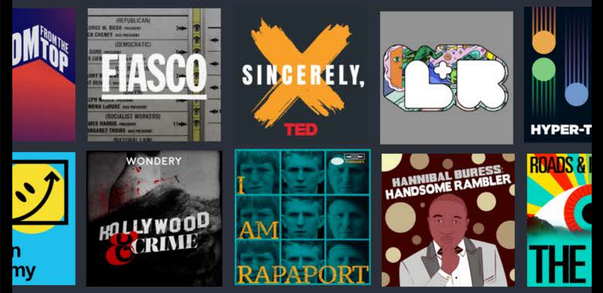

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | upgradeOffer |
|---|---|---|---|---|---|---|---|---|
| 90000003 | 11 | 1 | 0 | 2 | 352 | 0 | 1 | 0 |


9:41
Close
Keep exploring with us.
Thank you for being a loyal subscriber.
Get 1 month for free.
$2.49/mo for 6 months, then $4.99/mo.
Redeem your free month

9:41

# Luminary

PREMIUM




## Get 2 more months of Premium free!

As an early Premium subscriber we want to thank you by offering 2 more months of Premium on us. Enjoy!

Activate now

Followed by $7.99/month after

No Thanks

Payment will be charged to your Apple ID account at the confirmation of purchase. The subscription automatically renews unless it is canceled at least 24

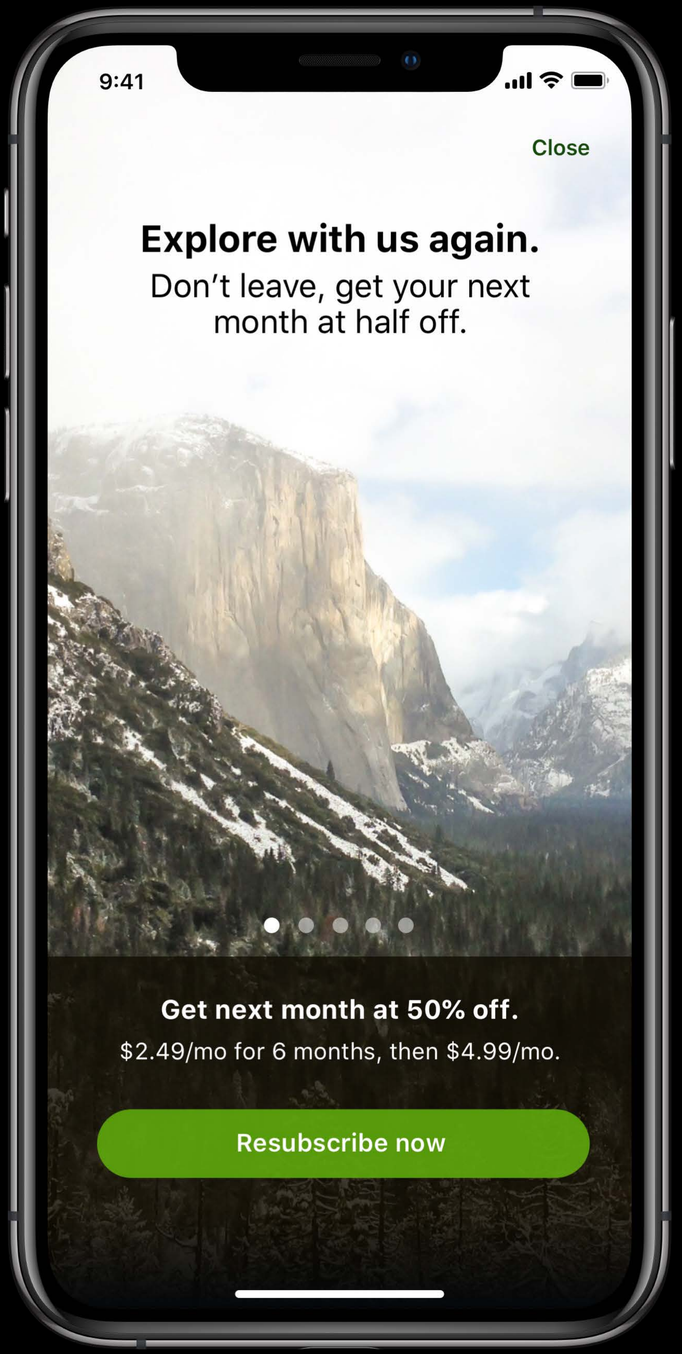
# Using Offers to Reduce Churn

## Save Offers


```
in_app: [{
  “purchase_date”: “2019-03-26…”,
  “expires_date”: “2019-04-26…”,…
}]
```

# Using Offers to Reduce Churn

## Save Offers


9:41
Close
Explore with us again.
Don’t leave, get your next month at half off.
Get next month at 50% off.
$2.49/mo for 6 months, then $4.99/mo.
Resubscribe now

# Using Offers to Reduce Churn

Reporting

NEW

# Using Offers to Reduce Churn

## Reporting

NEW

New reporting dashboards

- Subscription State
- Subscription Events

# Using Offers to Reduce Churn

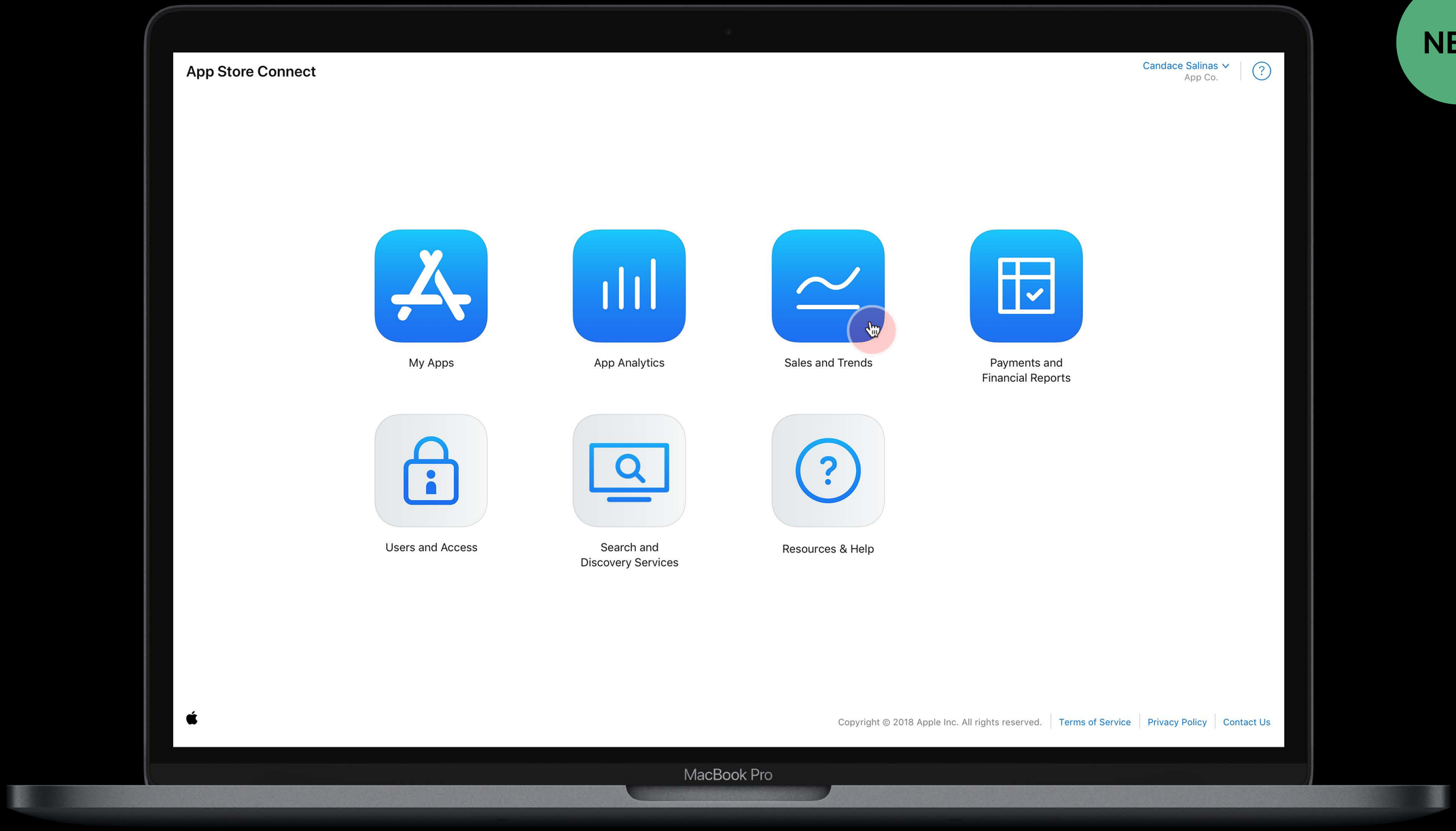
## Reporting

NEW

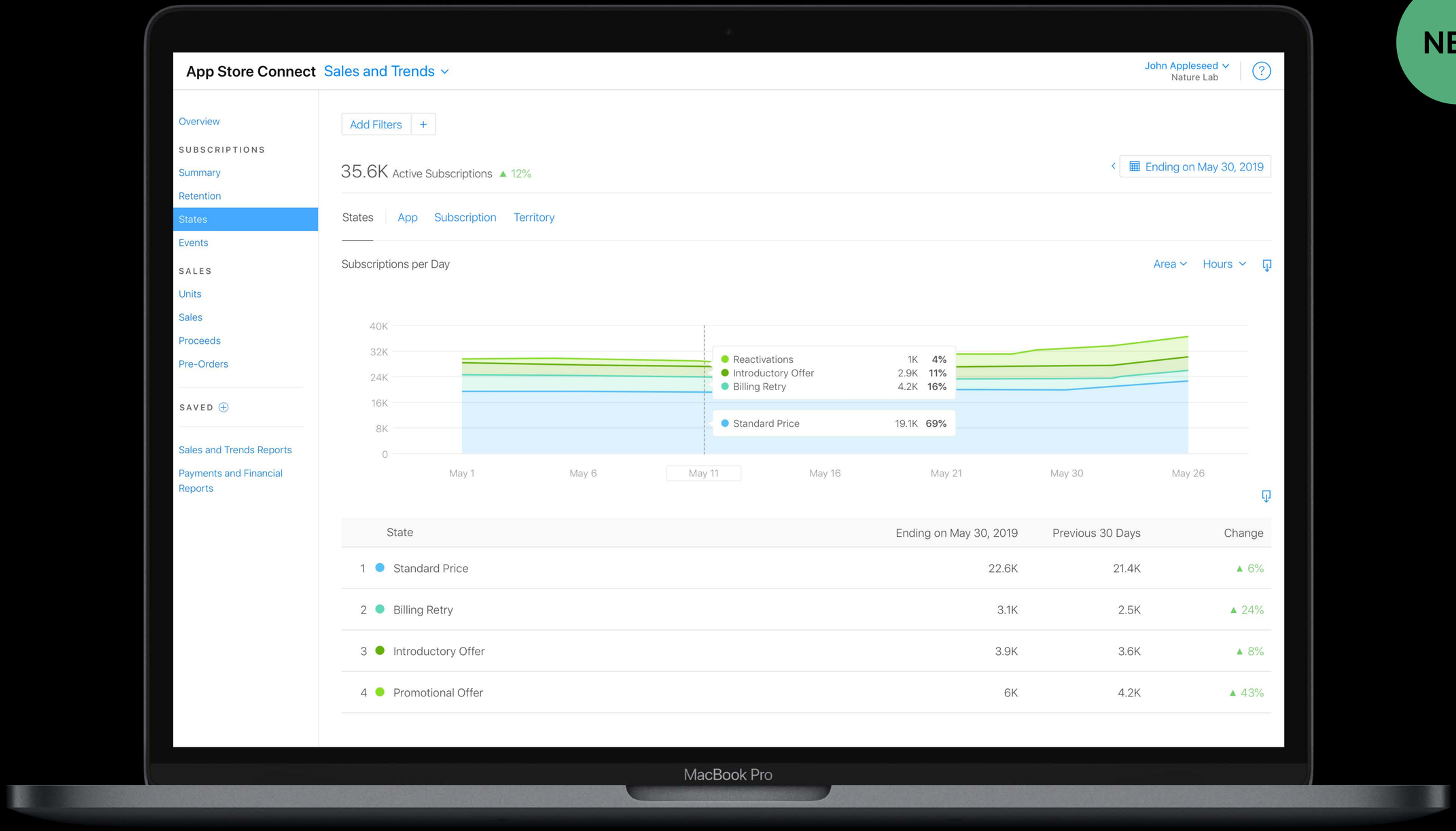
New reporting dashboards

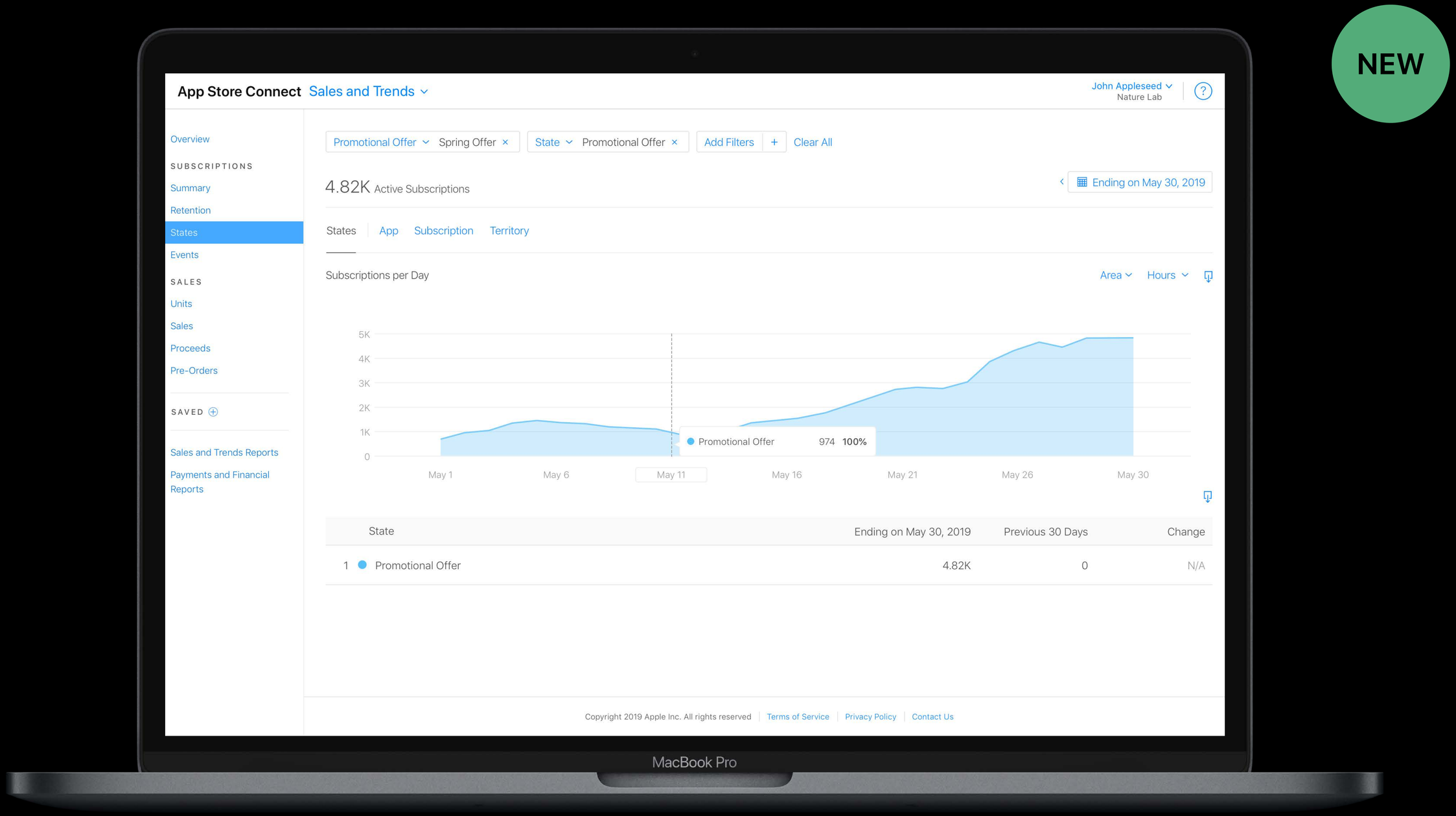
- Subscription State
- Subscription Events

Analyze subscription activity

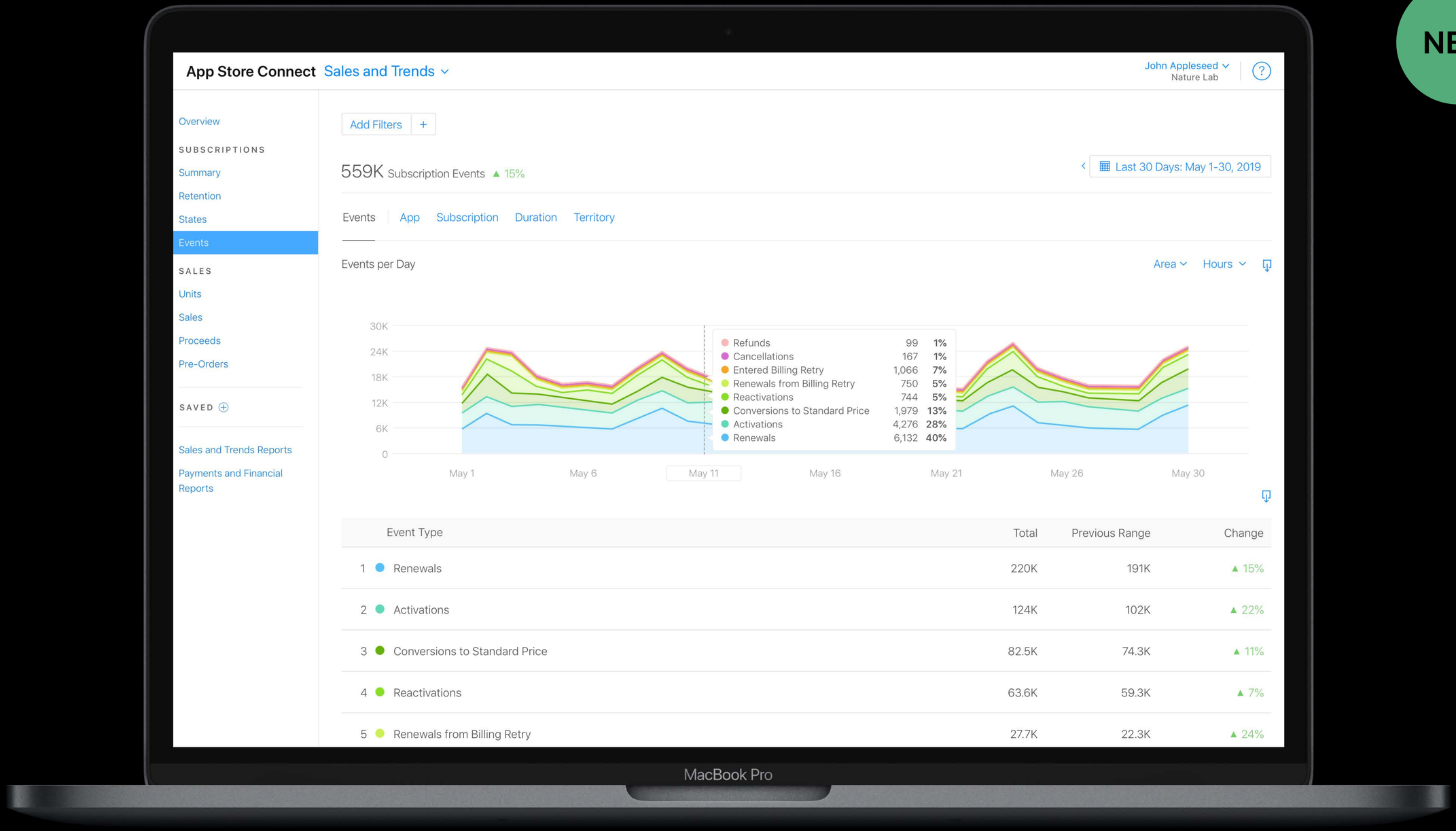

NEW
App Store Connect
Candace Salinas
App Co.
My Apps
App Analytics
Sales and Trends
Payments and Financial Reports
Users and Access
Search and Discovery Services
Resources & Help
Copyright © 2018 Apple Inc. All rights reserved.
Terms of Service
Privacy Policy
Contact Us
MacBook Pro

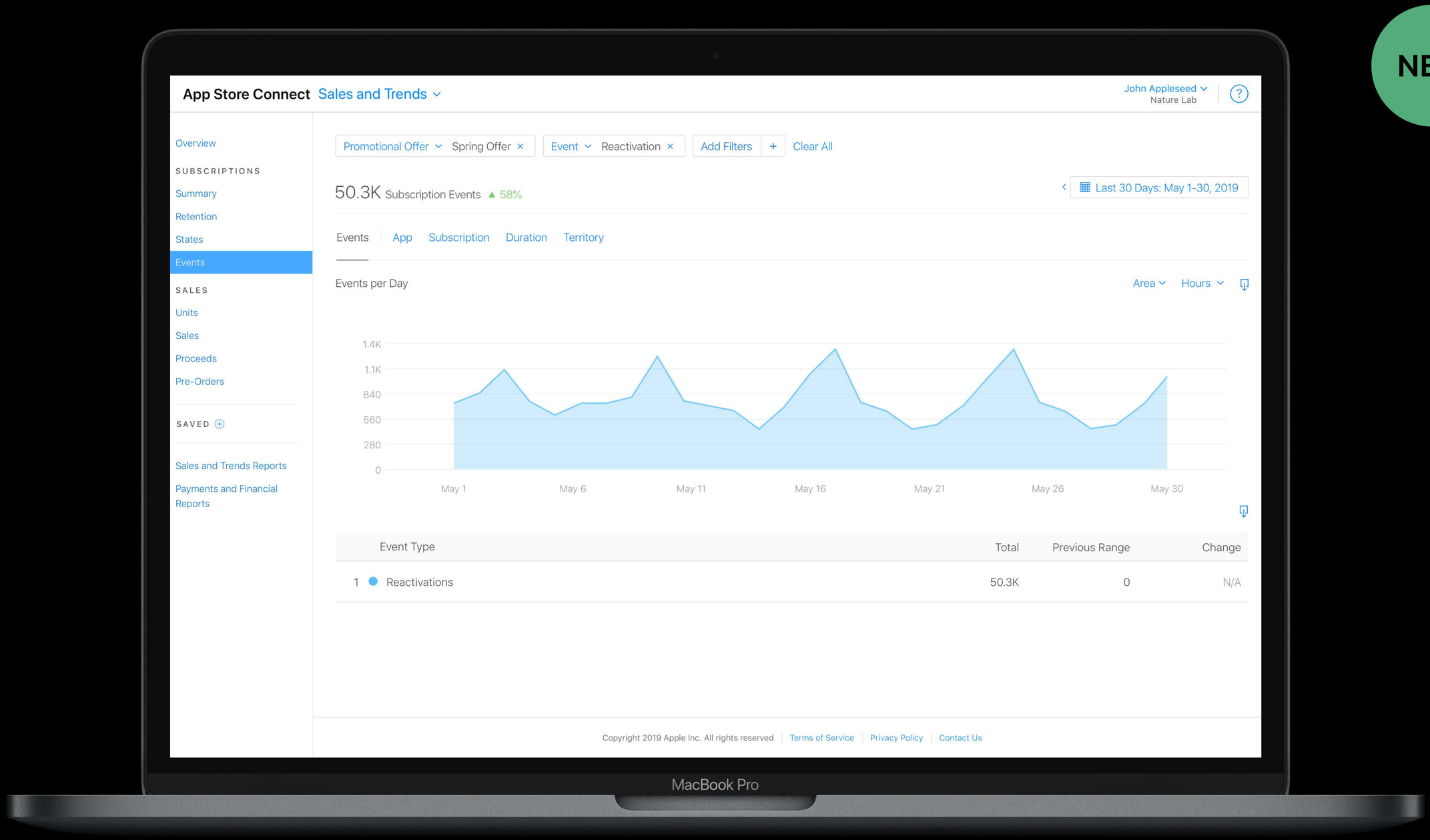
NEW


App Store Connect Sales and Trends
John Appleseed
Nature Lab
Overview
SUBSCRIPTIONS
Summary
Retention
States
Events
SALES
Units
Sales
Proceeds
Pre-Orders
SAVED
Sales and Trends Reports
Payments and Financial Reports
Add Filters
35.6K Active Subscriptions ▲ 12%
Ending on May 30, 2019
States
App
Subscription
Territory
Subscriptions per Day
Area
Hours
40K
32K
24K
16K
8K
0
Reactivations 1K 4%
Introductory Offer 2.9K 11%
Billing Retry 4.2K 16%
Standard Price 19.1K 69%
May 1
May 6
May 11
May 16
May 21
May 30
May 26
State
Ending on May 30, 2019
Previous 30 Days
Change
1 Standard Price 22.6K 21.4K ▲ 6%
2 Billing Retry 3.1K 2.5K ▲ 24%
3 Introductory Offer 3.9K 3.6K ▲ 8%
4 Promotional Offer 6K 4.2K ▲ 43%
MacBook Pro


NEW
App Store Connect
Sales and Trends
John Appleseed
Nature Lab
Overview
SUBSCRIPTIONS
Summary
Retention
States
Events
SALES
Units
Sales
Proceeds
Pre-Orders
SAVED
Sales and Trends Reports
Payments and Financial Reports
Promotional Offer
Spring Offer
State
Promotional Offer
Add Filters
Clear All
4.82K Active Subscriptions
Ending on May 30, 2019
States
App
Subscription
Territory
Subscriptions per Day
Area
Hours
5K
4K
3K
2K
1K
0
Promotional Offer 974 100%
May 1
May 6
May 11
May 16
May 21
May 26
May 30
State
Ending on May 30, 2019
Previous 30 Days
Change
1 Promotional Offer
4.82K
0
N/A
Copyright 2019 Apple Inc. All rights reserved
Terms of Service
Privacy Policy
Contact Us
MacBook Pro

NEW


App Store Connect
Sales and Trends
John Appleseed
Nature Lab
Overview
SUBSCRIPTIONS
Summary
Retention
States
Events
SALES
Units
Sales
Proceeds
Pre-Orders
SAVED
Sales and Trends Reports
Payments and Financial Reports
Add Filters
559K Subscription Events ▲ 15%
Last 30 Days: May 1-30, 2019
Events
App
Subscription
Duration
Territory
Events per Day
Area
Hours
30K
24K
18K
12K
6K
0
May 1
May 6
May 11
May 16
May 21
May 26
May 30
Refunds 99 1%
Cancellations 167 1%
Entered Billing Retry 1,066 7%
Renewals from Billing Retry 750 5%
Reactivations 744 5%
Conversions to Standard Price 1,979 13%
Activations 4,276 28%
Renewals 6,132 40%
Event Type
Total
Previous Range
Change
1 Renewals 220K 191K ▲ 15%
2 Activations 124K 102K ▲ 22%
3 Conversions to Standard Price 82.5K 74.3K ▲ 11%
4 Reactivations 63.6K 59.3K ▲ 7%
5 Renewals from Billing Retry 27.7K 22.3K ▲ 24%
MacBook Pro


NEW
App Store Connect
Sales and Trends
John Appleseed
Nature Lab
Overview
SUBSCRIPTIONS
Summary
Retention
States
Events
SALES
Units
Sales
Proceeds
Pre-Orders
SAVED
Sales and Trends Reports
Payments and Financial Reports
Promotional Offer
Spring Offer
Event
Reactivation
Add Filters
Clear All
50.3K Subscription Events ▲ 58%
Last 30 Days: May 1-30, 2019
Events
App
Subscription
Duration
Territory
Events per Day
Area
Hours
1.4K
1.1K
840
560
280
0
May 1
May 6
May 11
May 16
May 21
May 26
May 30
Event Type
Total
Previous Range
Change
1
Reactivations
50.3K
0
N/A
Copyright 2019 Apple Inc. All rights reserved
Terms of Service
Privacy Policy
Contact Us
MacBook Pro

# Summary

# Summary

Implement Subscription Offers

# Summary

Implement Subscription Offers

- In your app and on your server

# Summary

Implement Subscription Offers

- In your app and on your server

Implement customer eligibility

# Summary

Implement Subscription Offers

- In your app and on your server

Implement customer eligibility

- Receipt

# Summary

Implement Subscription Offers

- In your app and on your server

Implement customer eligibility

- Receipt
- Sever-to-server notifications

# Summary

Implement Subscription Offers

- In your app and on your server

Implement customer eligibility

- Receipt
- Sever-to-server notifications
- Developer data

# Summary

Implement Subscription Offers

- In your app and on your server

Implement customer eligibility

- Receipt
- Sever-to-server notifications
- Developer data

Present subscription offers to customers contextually

# Summary

Implement Subscription Offers

- In your app and on your server

Implement customer eligibility

- Receipt
- Sever-to-server notifications
- Developer data

Present subscription offers to customers contextually

Monitor performance of offers

# More Information

developer.apple.com/wwdc19/305

| | |
|---|---|
| In-App Purchases and Subscriptions Lab | Friday, 3:00 |


WWDC19