Workout experiences for watchOS and iOS with WorkoutKit

Audrey Sobgou Zebaze (@mvpohhhdrey)

🇨🇲

🇨🇲

gettyimages
Credit: Michael Regan

98729952

🇨🇲

🇨🇲

🇫🇷

🏉

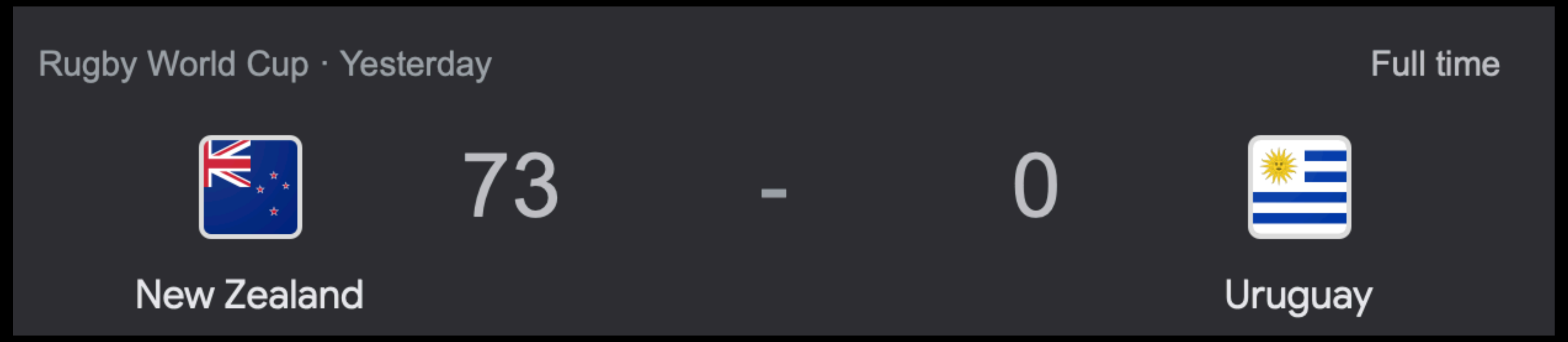
🇨🇲

🇫🇷

🇨🇲

🇫🇷

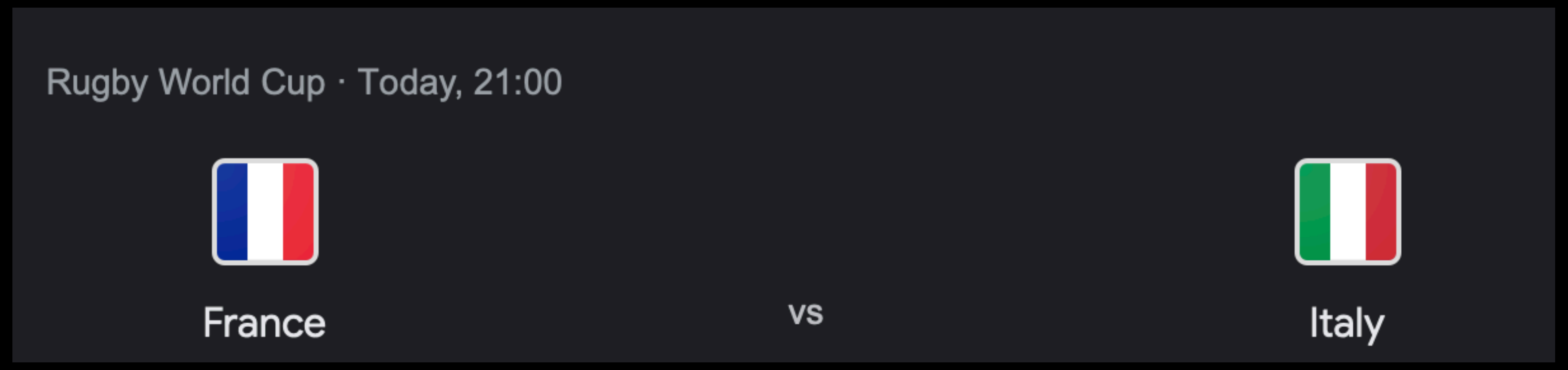
🏉



Rugby World Cup · Yesterday

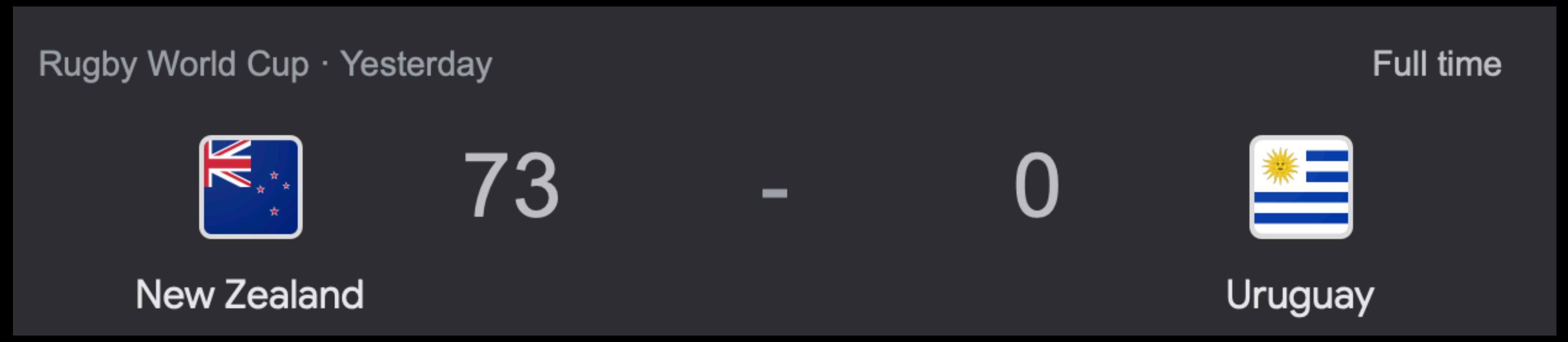
Full time

New Zealand	73	-	0	Uruguay



Rugby World Cup · Today, 21:00

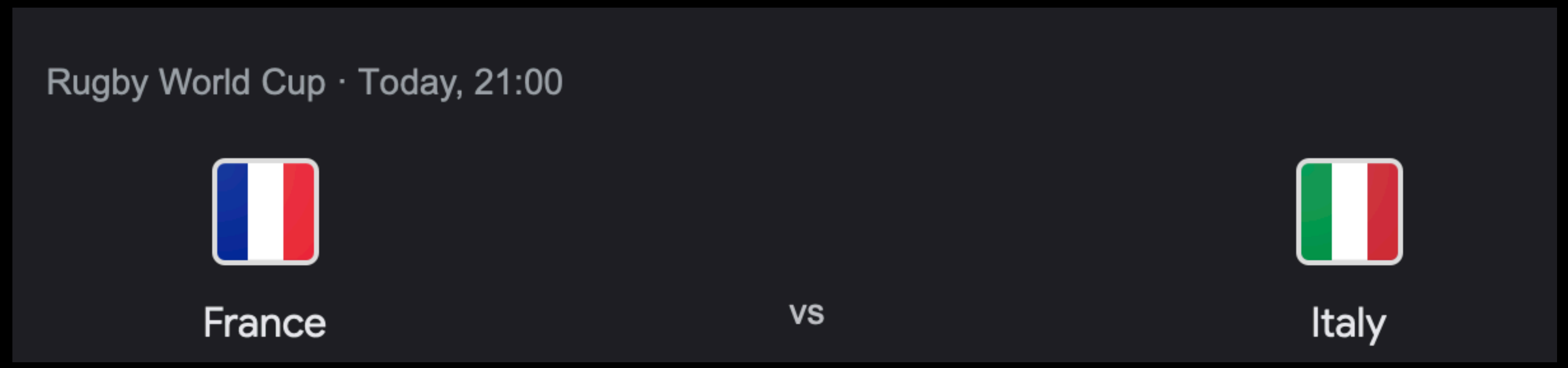
France vs Italy



Rugby World Cup · Yesterday

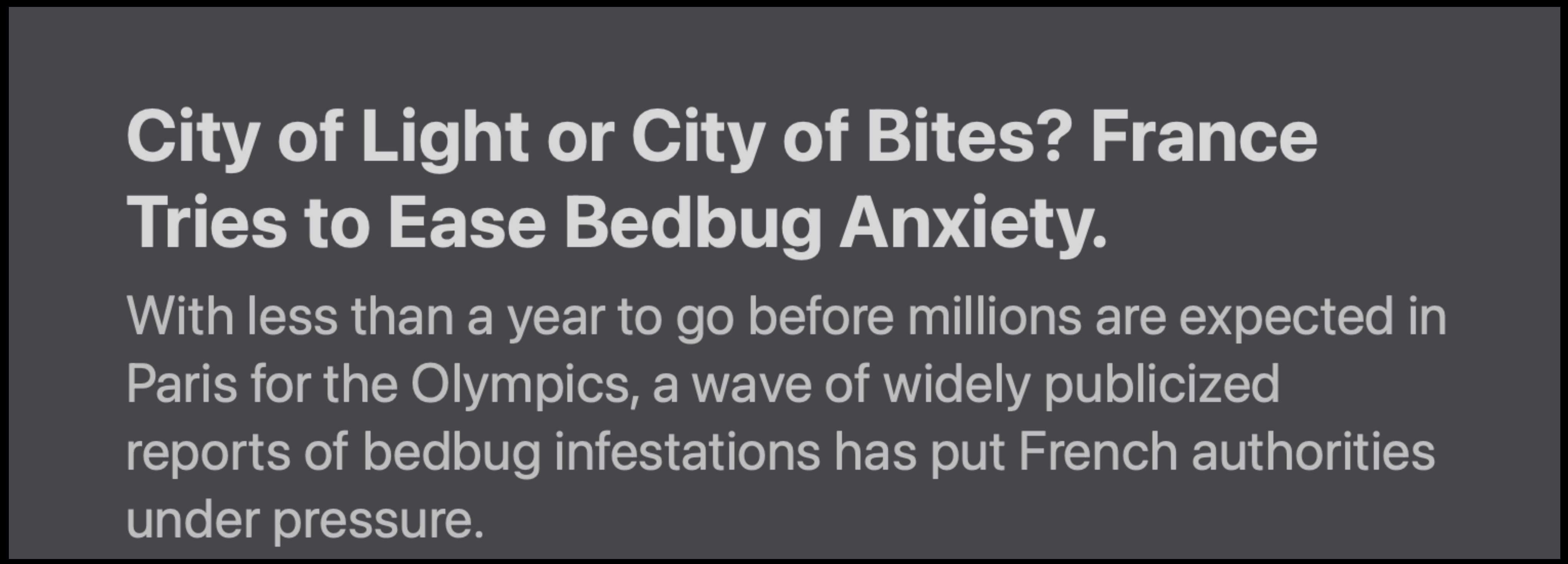
Full time

New Zealand 73 - 0 Uruguay



Rugby World Cup · Today, 21:00

France vs Italy



City of Light or City of Bites? France Tries to Ease Bedbug Anxiety.

With less than a year to go before millions are expected in Paris for the Olympics, a wave of widely publicized reports of bedbug infestations has put French authorities under pressure.

00101
02010
2
3
EDOX

Proton

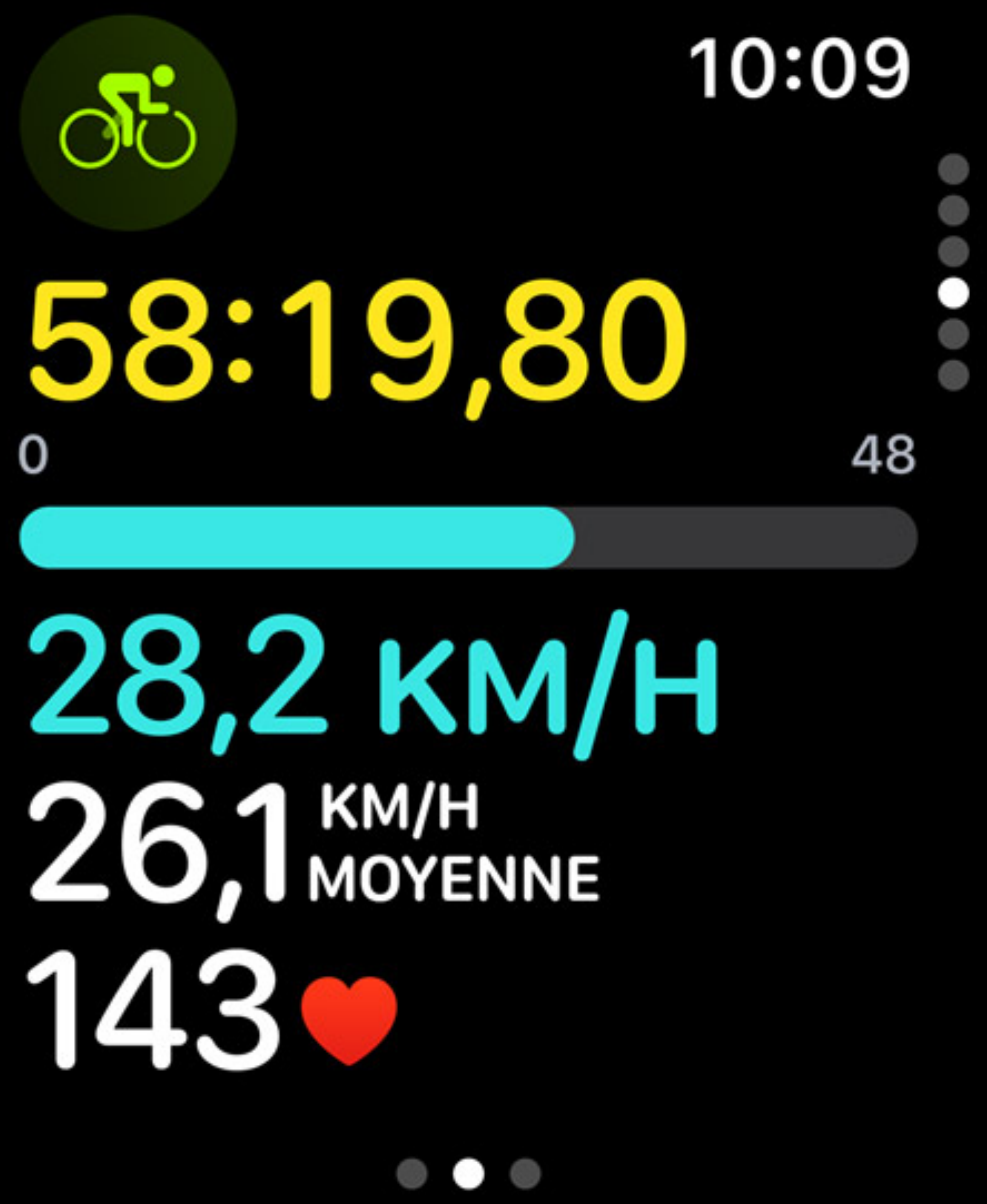
📁 💻

Why this talk?

- I ❤️ ⌚
- I ❤️ Sport
- I ❤️ tracking my performance
- I'll ❤️ to share Workout with current friends
- I would like to make new Apple Watch friends

Apple iPhone 6 - April 10, 2015

WorkoutKit



10:09

58:19,80

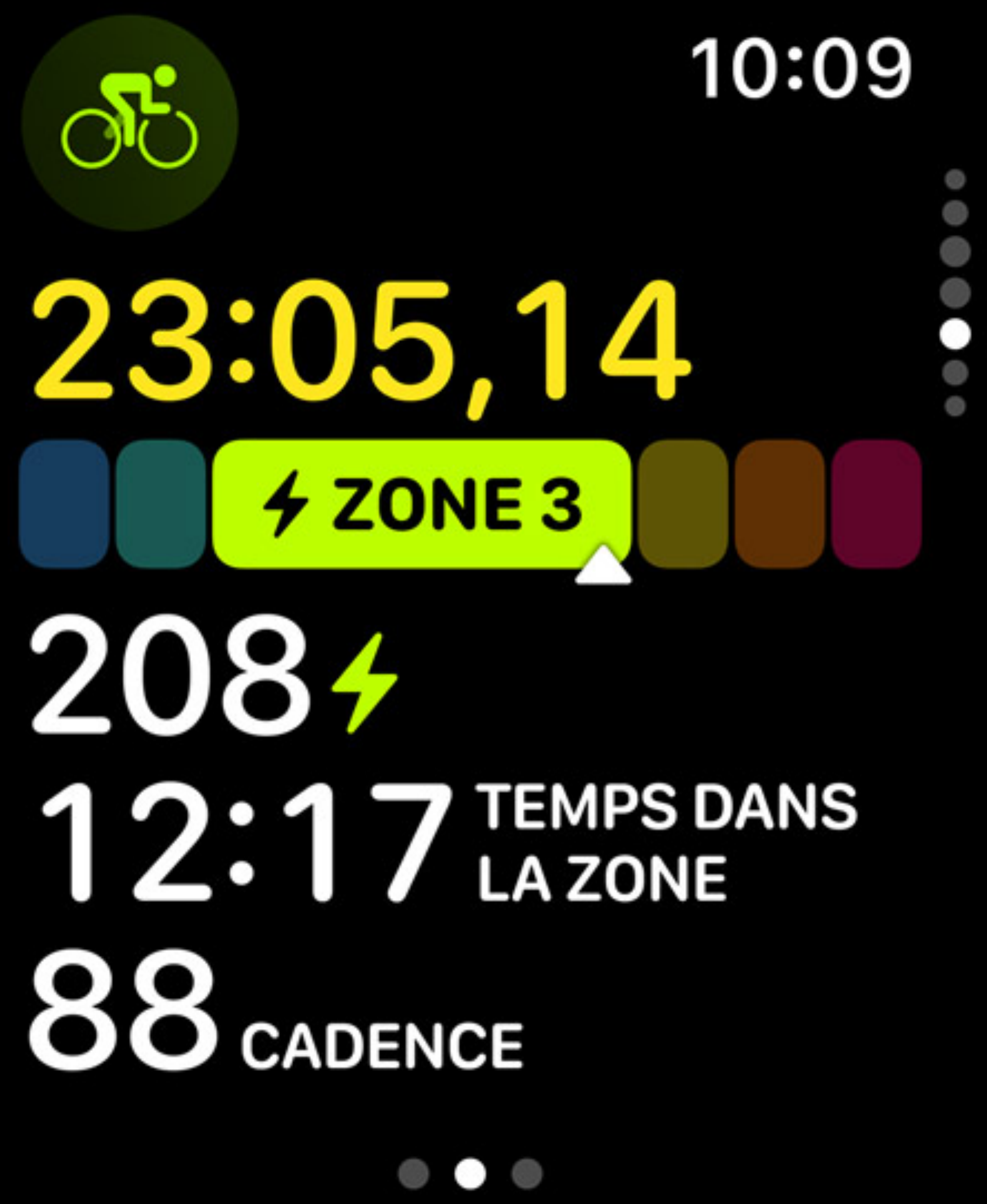
0

48

28,2 KM/H

26,1 KM/H MOYENNE

143 ❤



10:09

23:05,14

ZONE 3

208

12:17 TEMPS DANS LA ZONE

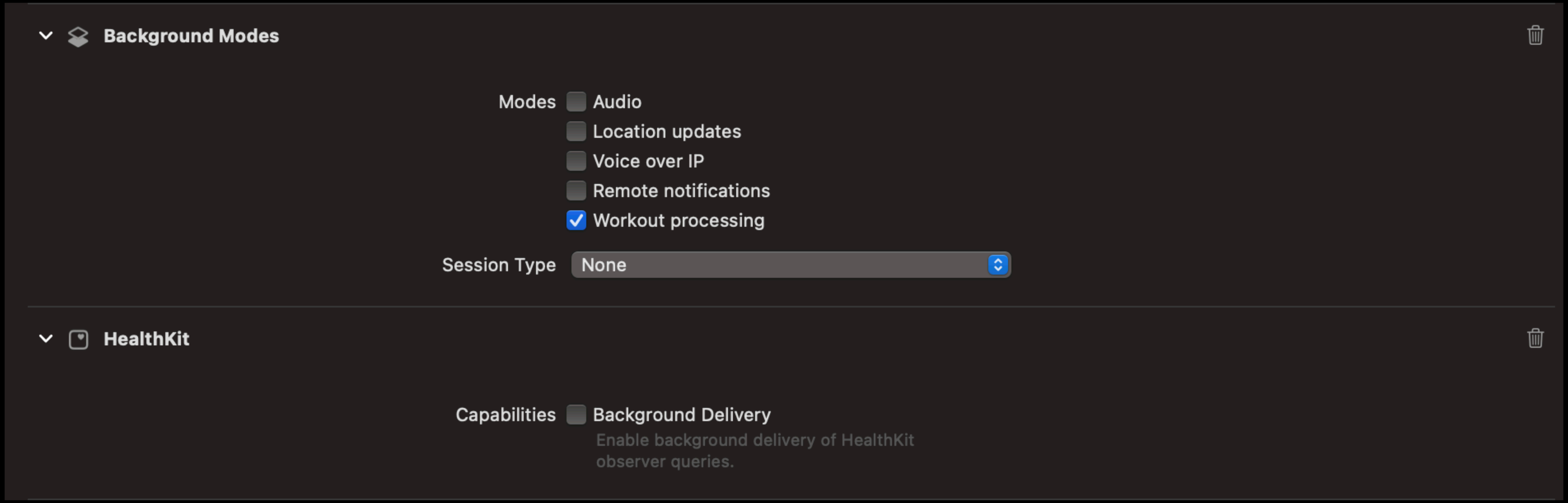
88 CADENCE

WorkoutKit Examples

- Workout Builder
- Workout Scheduler
- Workout Tracker

HealthKit

```
class HKHealthStore : NSObject
```



Background Modes

Modes
☐ Audio
☐ Location updates
☐ Voice over IP
☐ Remote notifications
☑ Workout processing

Session Type: None

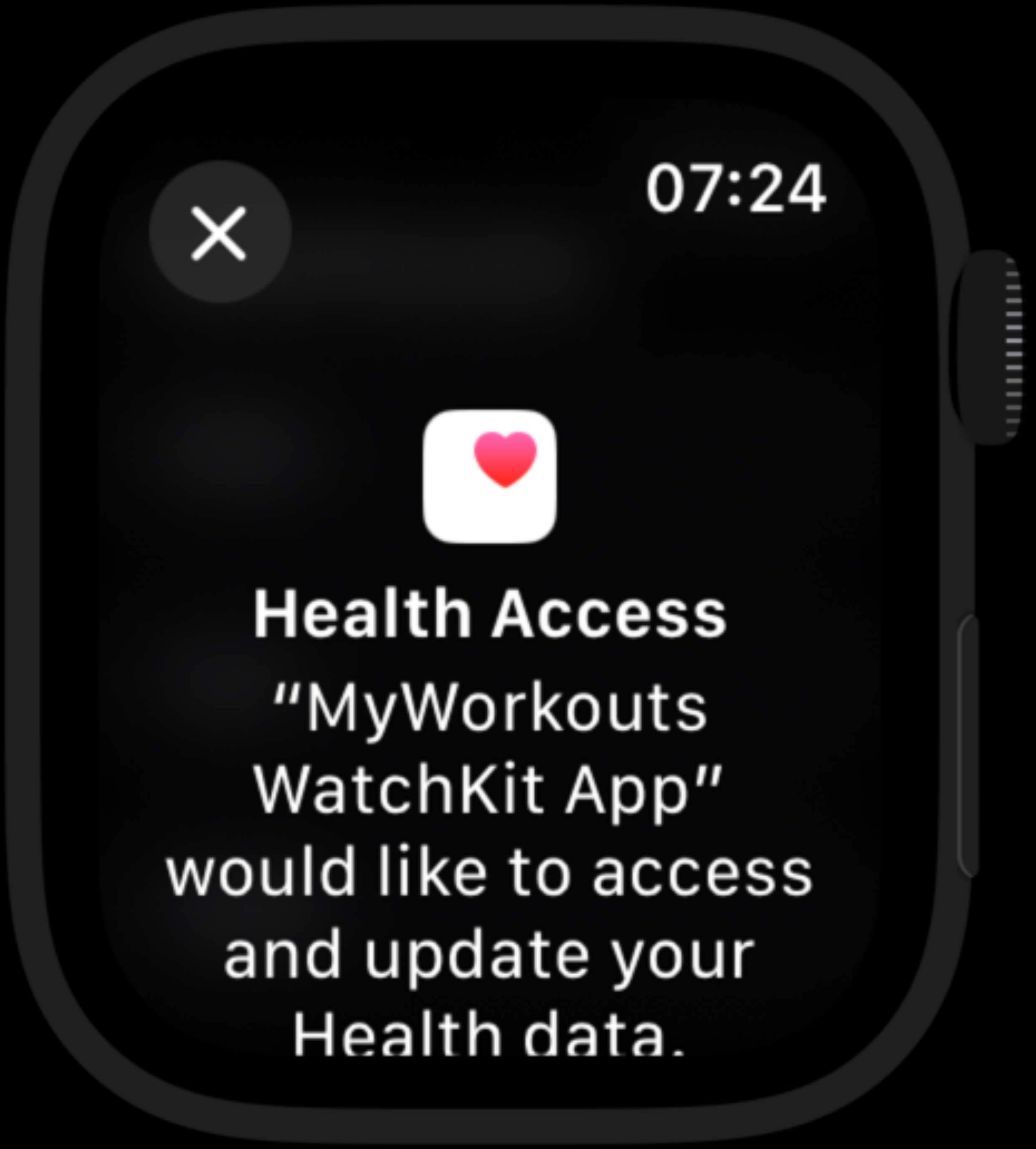
HealthKit

Capabilities
☐ Background Delivery
Enable background delivery of HealthKit observer queries.

Ensure HealthKit’s availability

```
import HealthKit

if HKHealthStore.isHealthDataAvailable() {
    // Add code to use HealthKit here.
}
```

Request Authorization

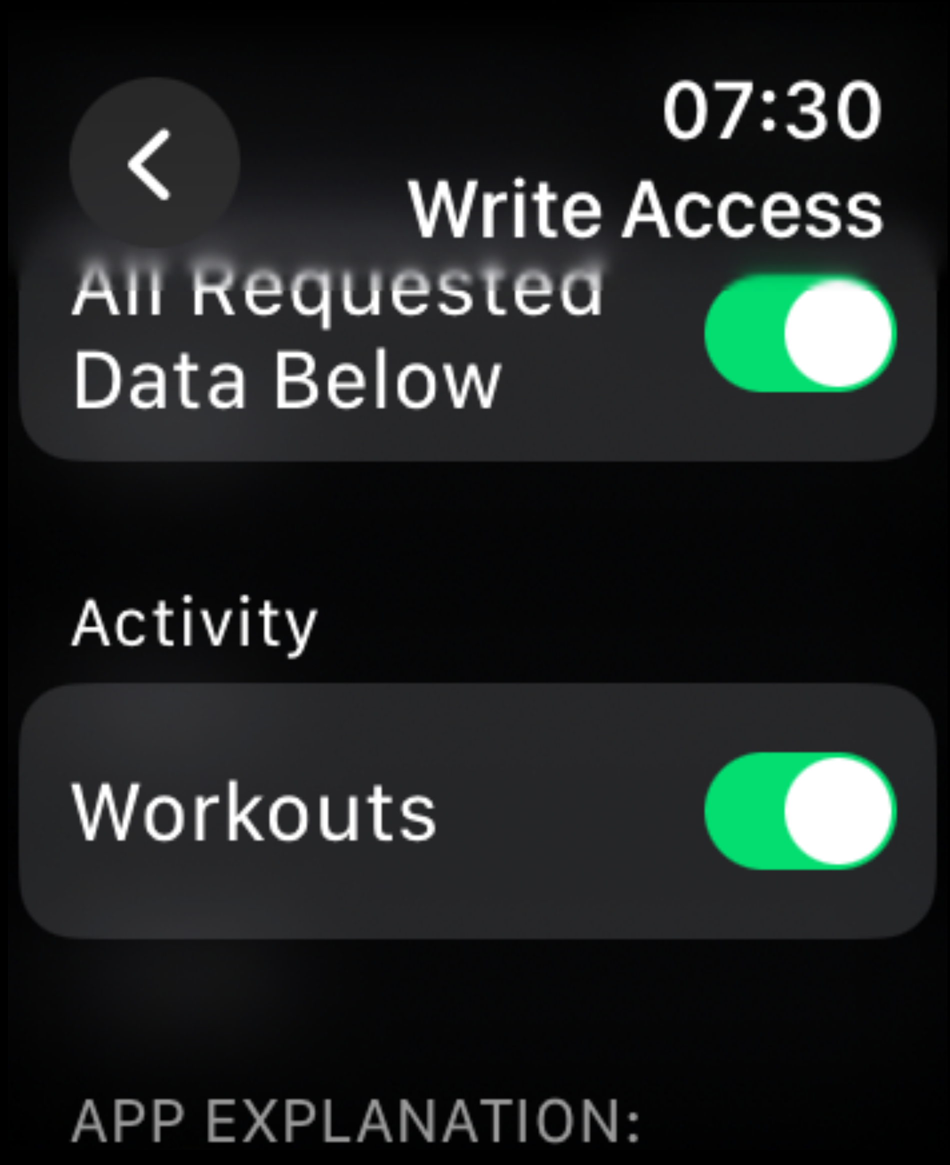


07:24

Health Access

"MyWorkouts WatchKit App" would like to access and update your Health data.

Request Authorization



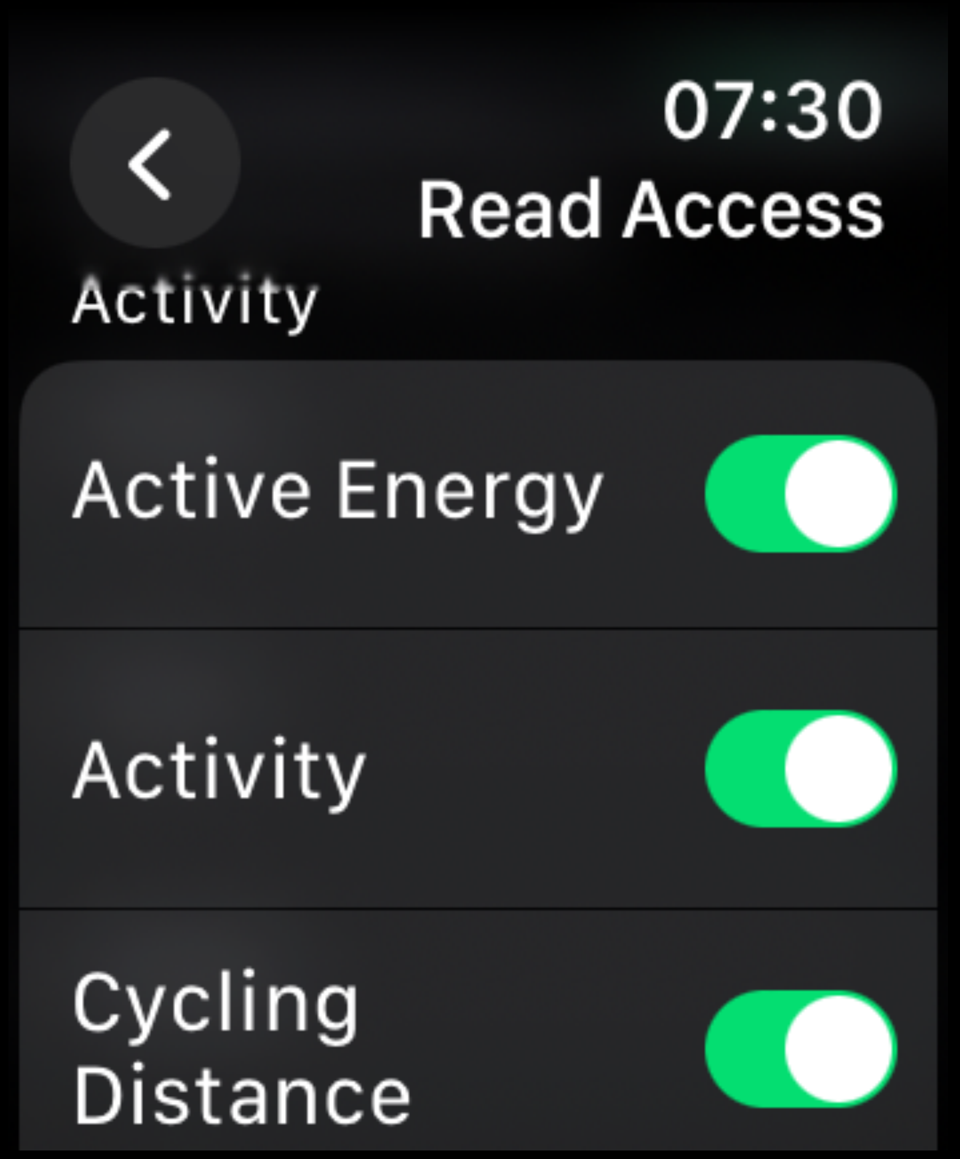
07:30
Write Access

All Requested Data Below ☑

Activity

Workouts ☑

APP EXPLANATION:



07:30
Read Access

Activity

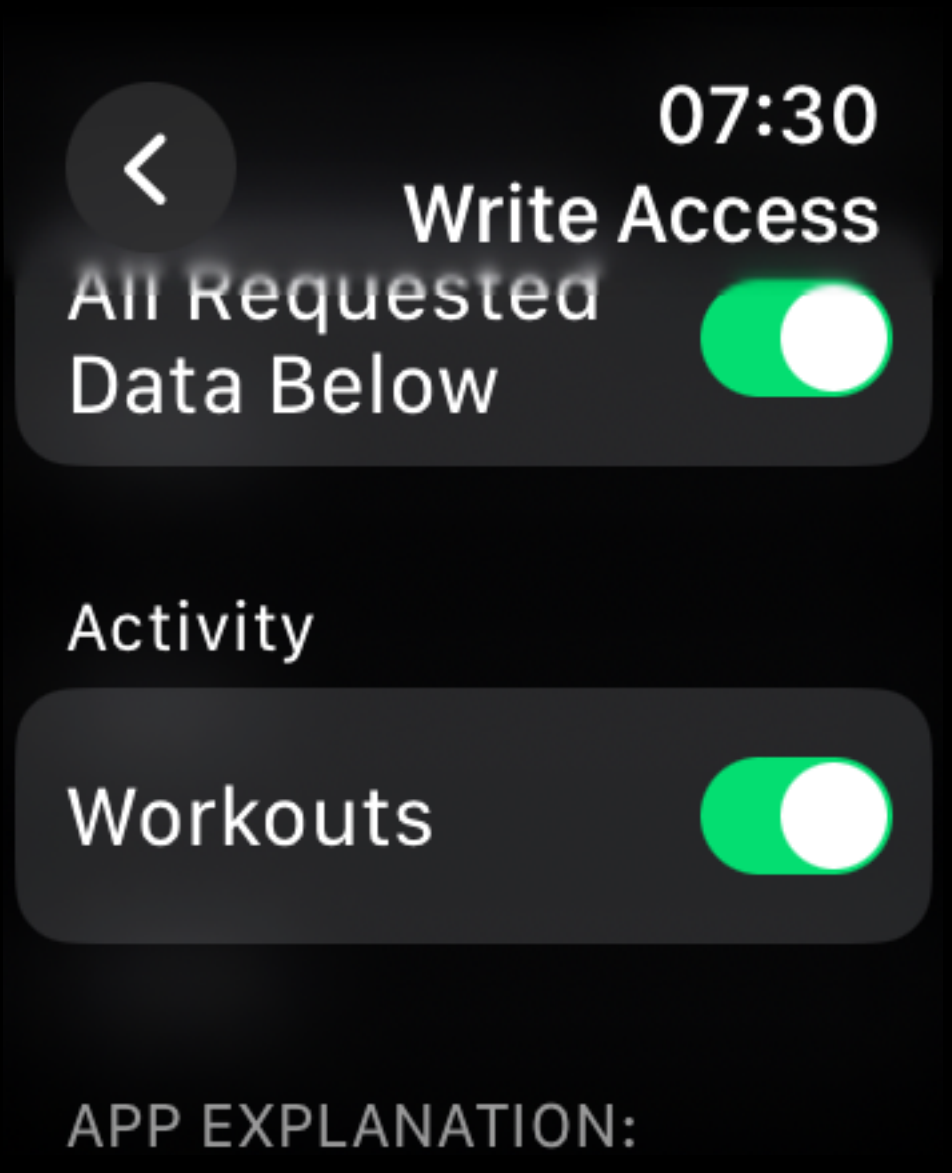
Active Energy ☑

Activity ☑

Cycling Distance ☑

Request Authorization

```
// Request authorization to access HealthKit.
func requestAuthorization() {
    // The quantity type to write to the health store.
    let typesToShare: Set = [
        HKQuantityType.workoutType()
    ]

    // The quantity types to read from the health store.
    let typesToRead: Set = [
      HKQuantityType(.heartRate),
      HKQuantityType(.activeEnergyBurned),
      HKQuantityType(.distanceWalkingRunning),
      HKQuantityType(.cyclingSpeed),
      HKQuantityType(.cyclingPower),
      HKQuantityType(.cyclingCadence),
      HKQuantityType(.distanceCycling),
      HKQuantityType.workoutType(),
      HKObjectType.activitySummaryType()
    ]

}
```



07:30

Write Access

All Requested Data Below: on

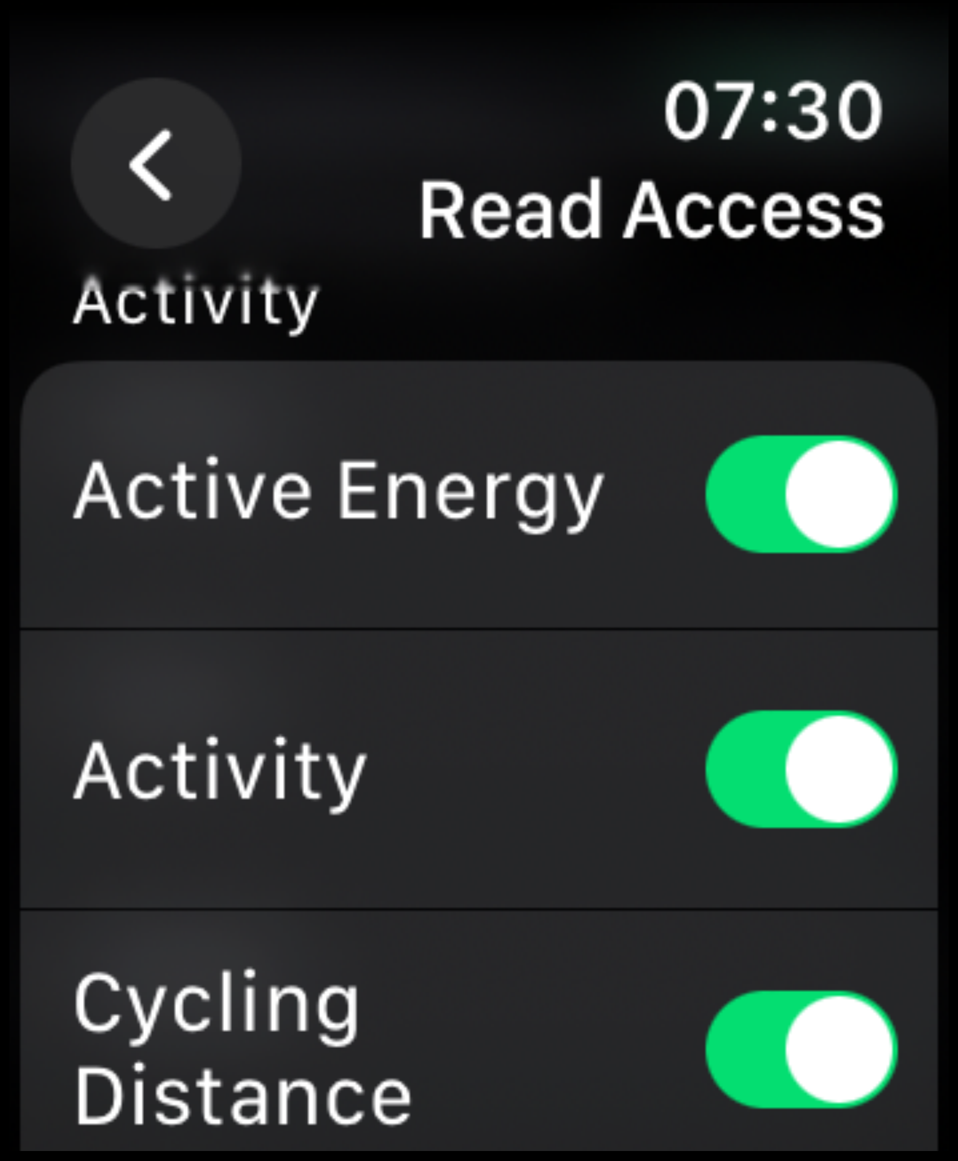
Activity

Workouts: on

APP EXPLANATION:

Request Authorization

```swift
// Request authorization to access HealthKit.
func requestAuthorization() {
    // The quantity type to write to the health store.
    let typesToShare: Set = [
        HKQuantityType.workoutType()
    ]

    // The quantity types to read from the health store.
    let typesToRead: Set = [
      HKQuantityType(.heartRate),
      HKQuantityType(.activeEnergyBurned),
      HKQuantityType(.distanceWalkingRunning),
      HKQuantityType(.cyclingSpeed),
      HKQuantityType(.cyclingPower),
      HKQuantityType(.cyclingCadence),
      HKQuantityType(.distanceCycling),
      HKQuantityType.workoutType(),
      HKObjectType.activitySummaryType()
    ]

}
```



07:30

Read Access

Activity

Active Energy ●

Activity ●

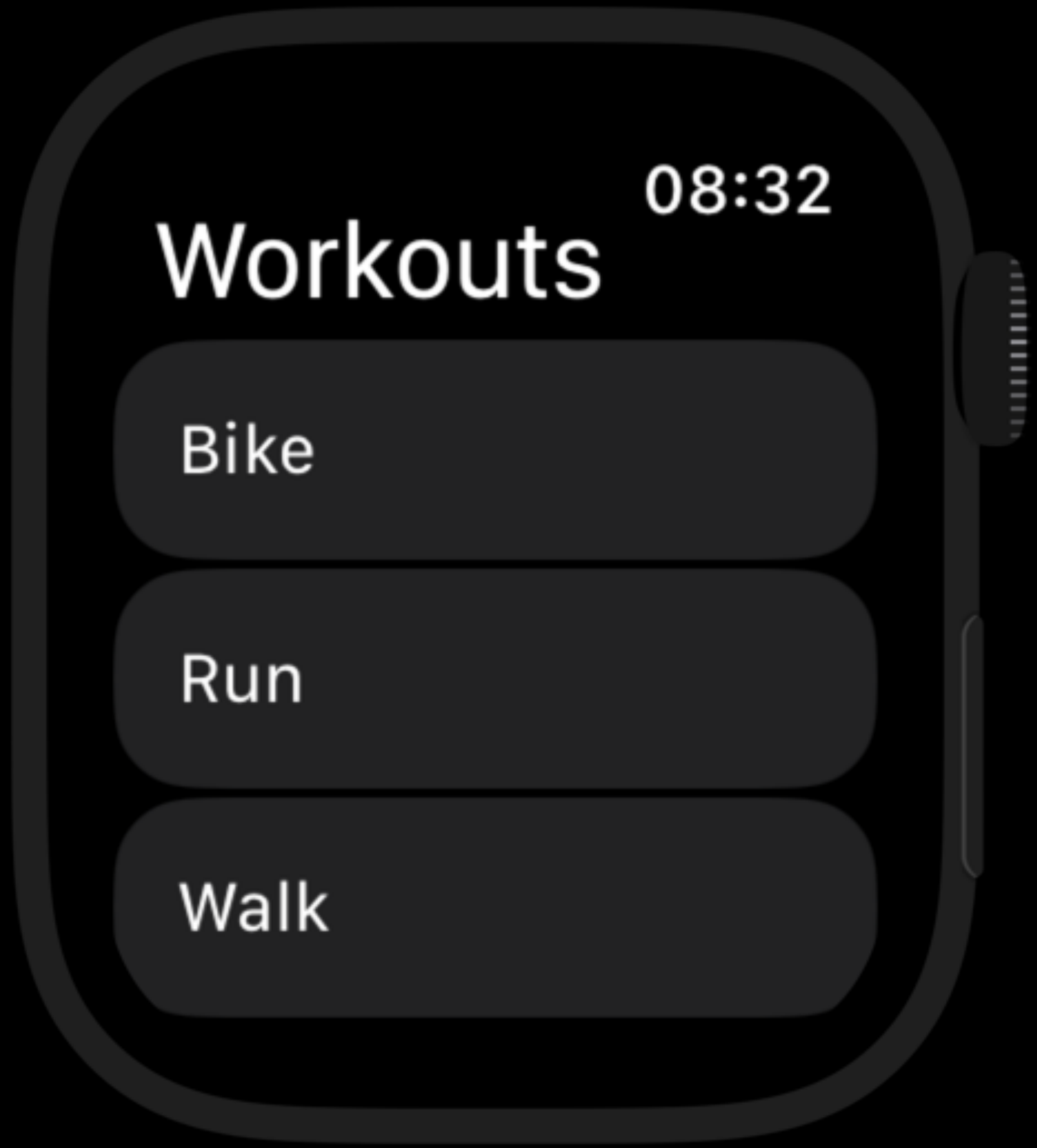
Cycling Distance ●

Request Authorization

```
func requestAuthorization() {
         …

         Task {
            do {
                try await healthStore.requestAuthorization(
                    toShare: typesToShare,
                    read: typesToRead
                )
            } catch {
                Logger.shared.log("Failed to request authorization: \(error)")
            }
          }
    }
```

Activities

```
@available(watchOS 2.0, *)
public enum HKWorkoutActivityType : UInt, @unchecked Sendable {

    case basketball = 6

    case curling = 12

    @available(watchOS 6.0, *)
    case discSports = 75

    @available(watchOS 9.0, *)
    case swimBikeRun = 82

    @available(watchOS 9.0, *)
    case transition = 83

    @available(watchOS 10.0, *)
    case underwaterDiving = 84


    case other = 3000
}
```



Create the Workout Session and Live Workout Builder

```
let configuration = HKWorkoutConfiguration()
configuration.activityType = workoutType
configuration.locationType = .outdoor
```

Create the Workout Session and Live Workout Builder

```
var session: HKWorkoutSession?
#if os(watchOS)
/**
The live workout builder that is only available on watchOS.
*/
var builder: HKLiveWorkoutBuilder?

func startWorkout(workoutConfiguration: HKWorkoutConfiguration) async throws {

        session = try HKWorkoutSession(healthStore: healthStore, configuration: workoutConfiguration)
        builder = session?.associatedWorkoutBuilder()
        session?.delegate = self
        builder?.delegate = self
        builder?.dataSource = HKLiveWorkoutDataSource(healthStore: healthStore, workoutConfiguration:
workoutConfiguration)
        /**
          Start mirroring the session to the companion device.
         */
        try await session?.startMirroringToCompanionDevice()
        /**
          Start the workout session activity.
         */
        let startDate = Date()
        session?.startActivity(with: startDate)
        try await builder?.beginCollection(at: startDate)
    }
```

Create the Workout Session and Live Workout Builder

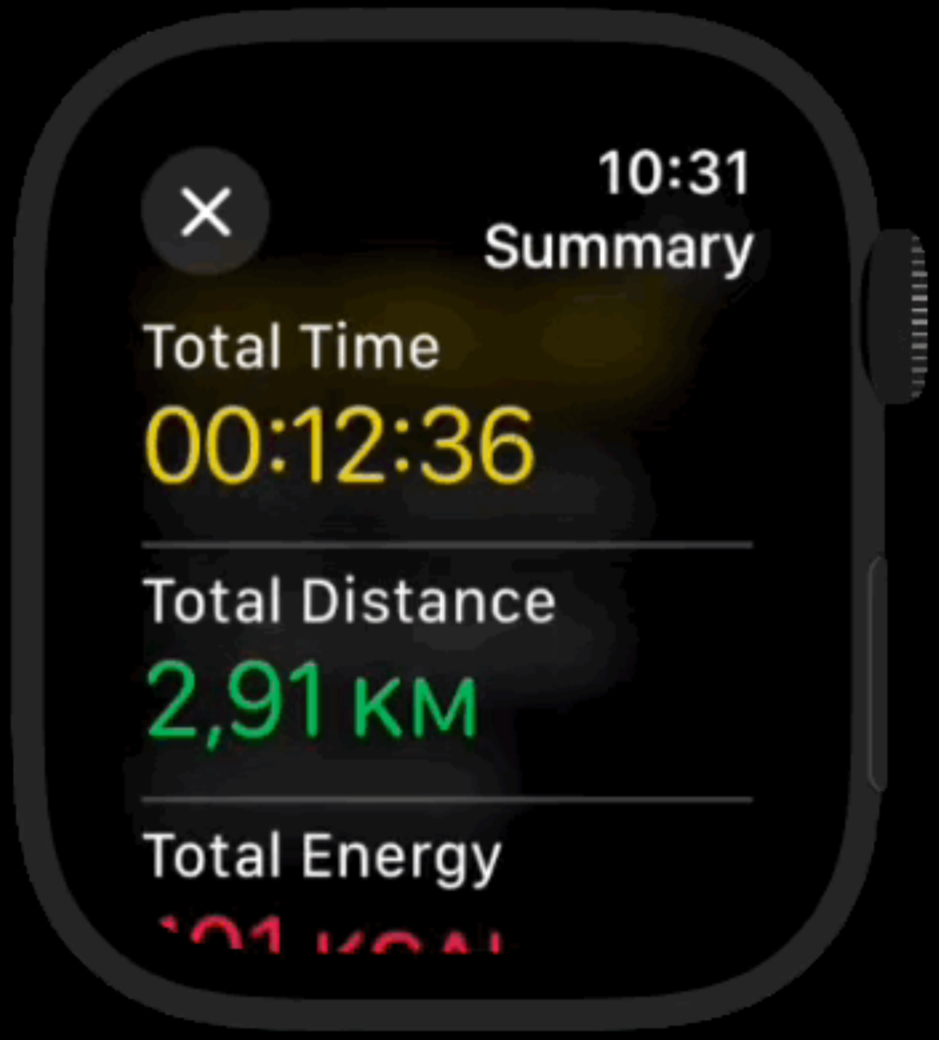
```swift
var session: HKWorkoutSession?
#if os(watchOS)
/**
The live workout builder that is only available on watchOS.
*/
var builder: HKLiveWorkoutBuilder?

func startWorkout(workoutConfiguration: HKWorkoutConfiguration) async throws {

        session = try HKWorkoutSession(healthStore: healthStore, configuration: workoutConfiguration)
        builder = session?.associatedWorkoutBuilder()
        session?.delegate = self
        builder?.delegate = self
        builder?.dataSource = HKLiveWorkoutDataSource(healthStore: healthStore, workoutConfiguration:
workoutConfiguration)
        /**
          Start mirroring the session to the companion device.
         */
        try await session?.startMirroringToCompanionDevice()
        /**
          Start the workout session activity.
         */
        let startDate = Date()
        session?.startActivity(with: startDate)
        try await builder?.beginCollection(at: startDate)
    }
```

Create the Workout Session and Live Workout Builder

```
var session: HKWorkoutSession?
#if os(watchOS)
/**
The live workout builder that is only available on watchOS.
*/
var builder: HKLiveWorkoutBuilder?

func startWorkout(workoutConfiguration: HKWorkoutConfiguration) async throws {

        session = try HKWorkoutSession(healthStore: healthStore, configuration: workoutConfiguration)
        builder = session?.associatedWorkoutBuilder()
        session?.delegate = self
        builder?.delegate = self
        builder?.dataSource = HKLiveWorkoutDataSource(healthStore: healthStore, workoutConfiguration:
workoutConfiguration)
        /**
          Start mirroring the session to the companion device.
         */
        try await session?.startMirroringToCompanionDevice()
        /**
          Start the workout session activity.
         */
        let startDate = Date()
        session?.startActivity(with: startDate)
        try await builder?.beginCollection(at: startDate)
    }
```

Create the Workout Session and Live Workout Builder

```
var session: HKWorkoutSession?
#if os(watchOS)
/**
The live workout builder that is only available on watchOS.
*/
var builder: HKLiveWorkoutBuilder?

func startWorkout(workoutConfiguration: HKWorkoutConfiguration) async throws {

        session = try HKWorkoutSession(healthStore: healthStore, configuration: workoutConfiguration)
        builder = session?.associatedWorkoutBuilder()
        session?.delegate = self
        builder?.delegate = self
        builder?.dataSource = HKLiveWorkoutDataSource(healthStore: healthStore, workoutConfiguration:
workoutConfiguration)
        /**
          Start mirroring the session to the companion device.
         */
        try await session?.startMirroringToCompanionDevice()
        /**
          Start the workout session activity.
         */
        let startDate = Date()
        session?.startActivity(with: startDate)
        try await builder?.beginCollection(at: startDate)
    }
```



10:31
Summary

Total Time
00:12:36

Total Distance
2,91 KM

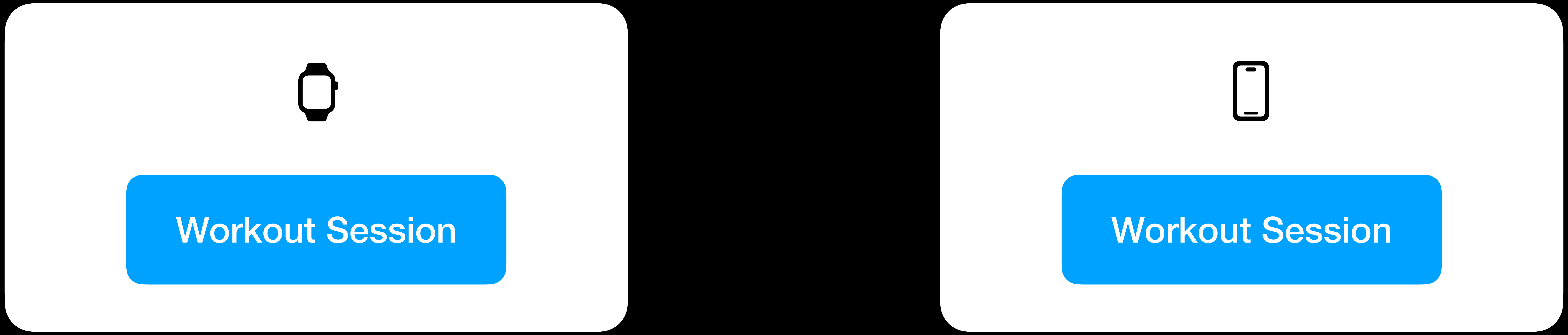
Total Energy
101 KCAL

Mirroring

Workout Session

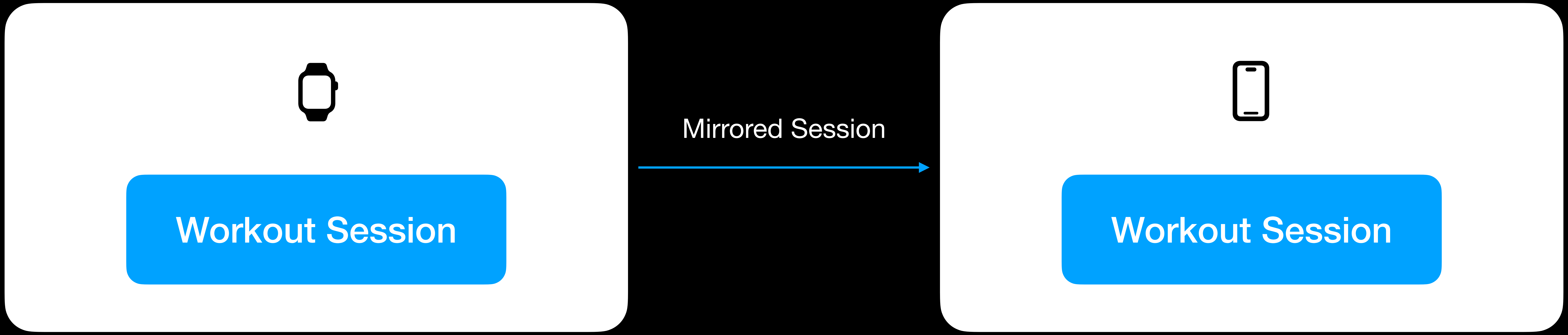
⌚

Workout Session



Workout Session

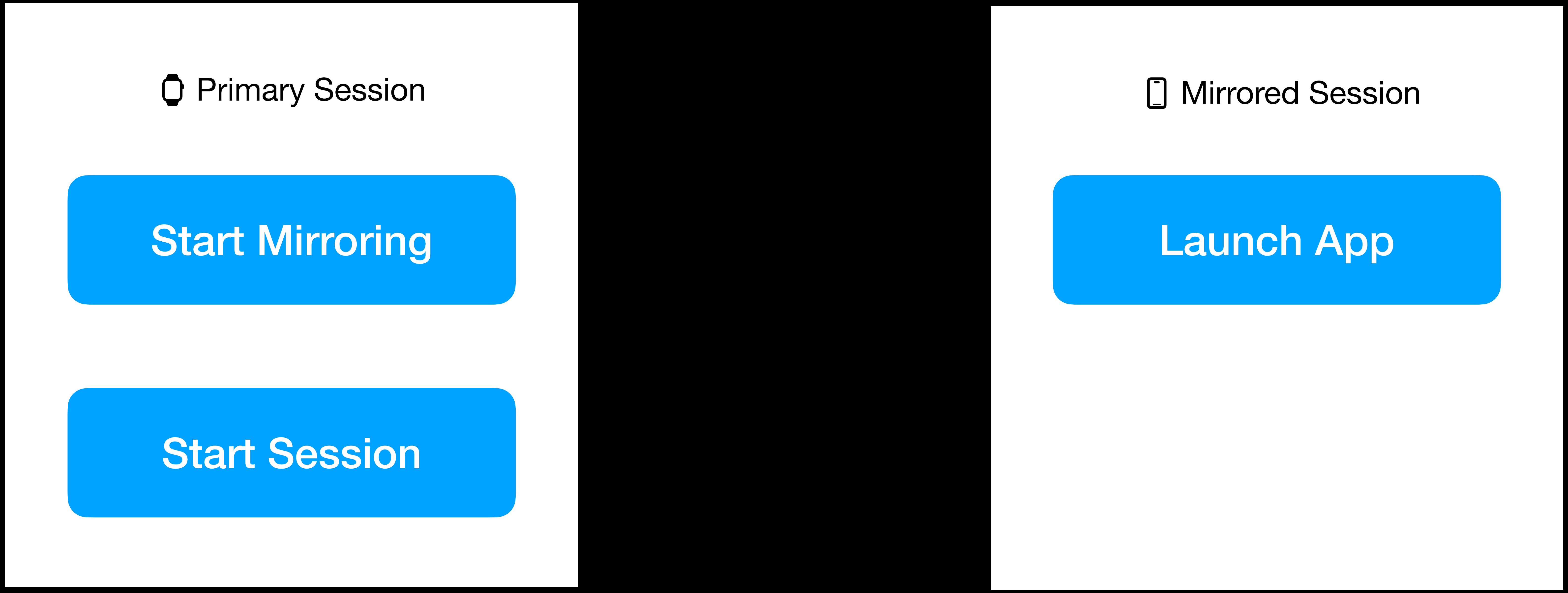
Workout Session



Workout Session

Mirrored Session

Workout Session



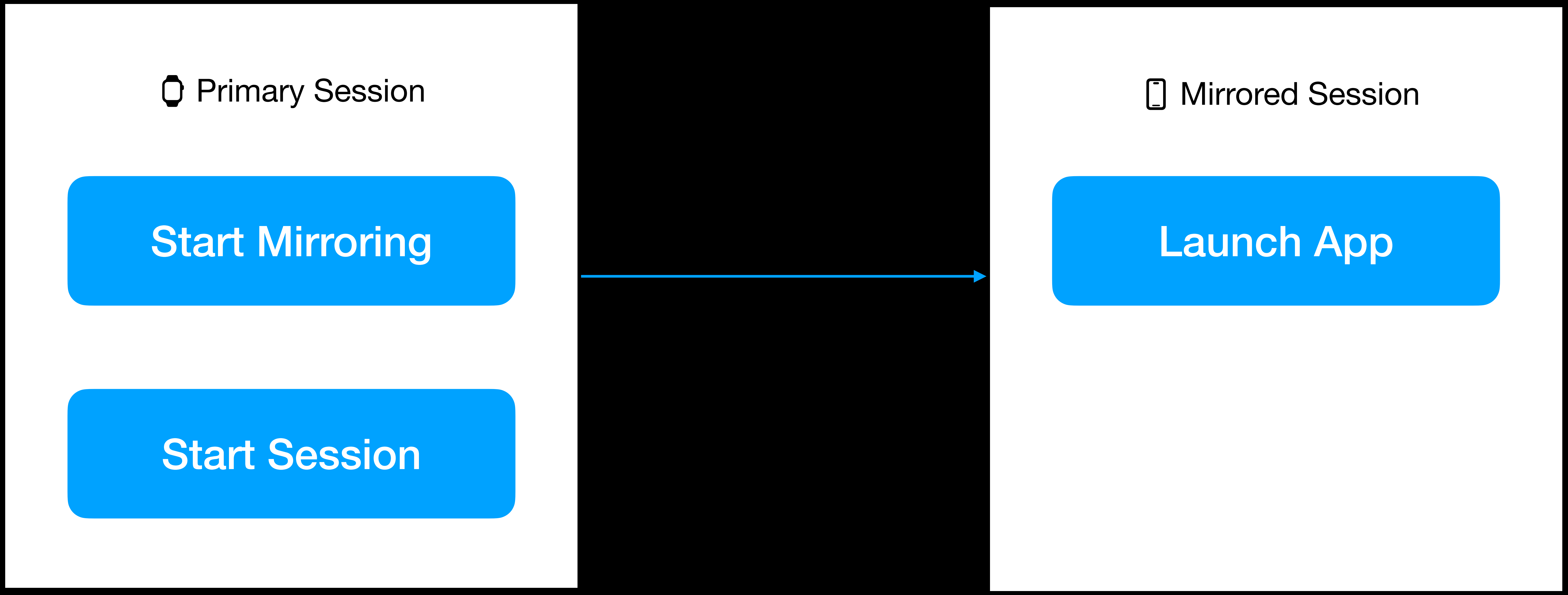
⌚ Primary Session

Start Mirroring

Start Session

📱 Mirrored Session

Launch App



Primary Session

Start Mirroring

Start Session

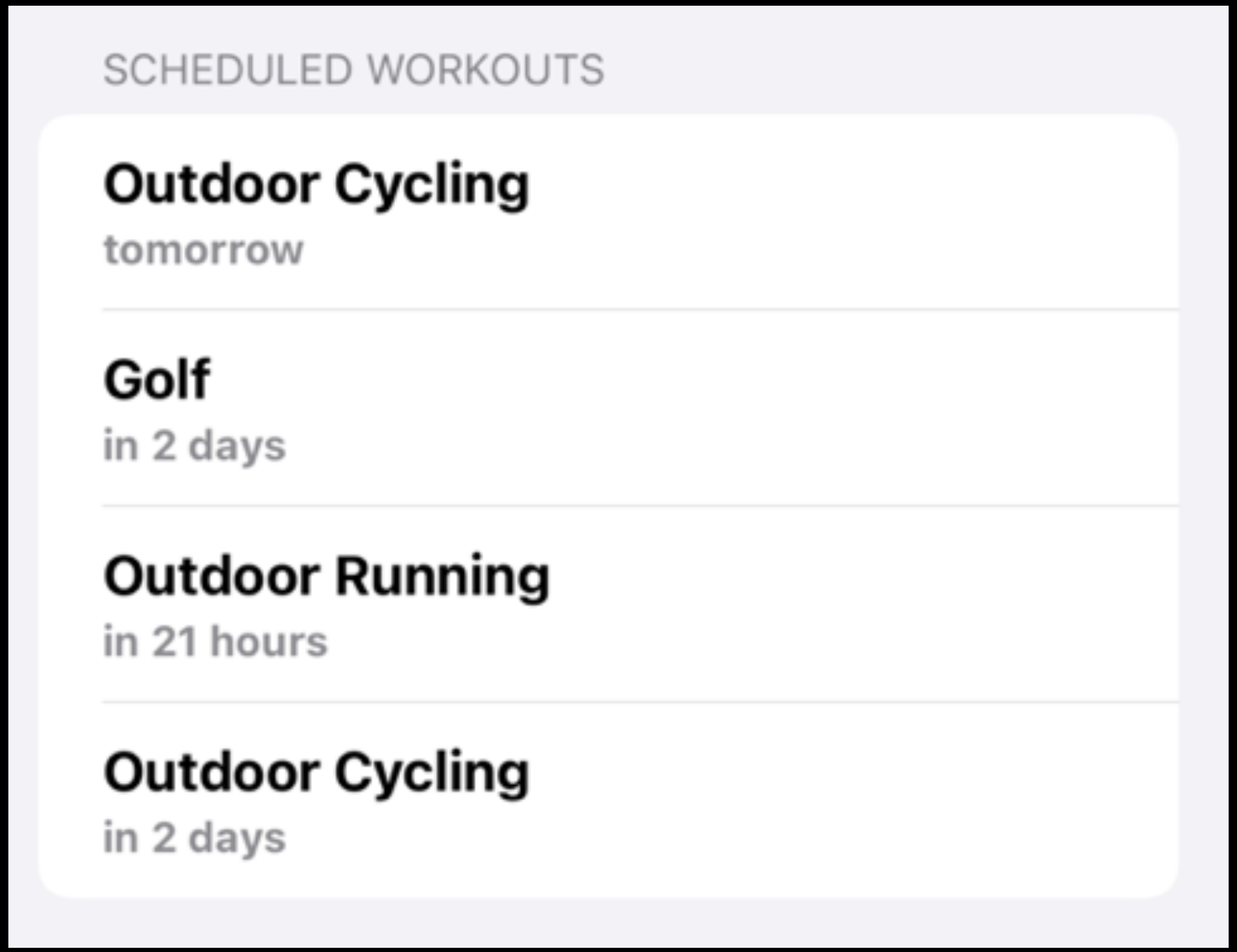
Mirrored Session

Launch App

Mirroring

```
    func retrieveRemoteSession() {
        /**
         HealthKit calls this handler when a session starts mirroring.
         */
        healthStore.workoutSessionMirroringStartHandler = { mirroredSession in
            Task {
                self.resetWorkout()
                self.session = mirroredSession
                self.session?.delegate = self
                Logger.shared.log("Start mirroring remote session: \.
(mirroredSession)”)
            }
        }
    }
```

Planning



SCHEDULED WORKOUTS

Outdoor Cycling
tomorrow

Golf
in 2 days

Outdoor Running
in 21 hours

Outdoor Cycling
in 2 days

WorkoutKit

Planning

```
final public class WorkoutScheduler {

    public static let shared: WorkoutScheduler

    public static let maxAllowedScheduledWorkoutCount: Int

    public static var isSupported: Bool { get }

    final public var scheduledWorkouts: [ScheduledWorkoutPlan] { get async }

    final public func schedule(_ workout: WorkoutPlan, at: DateComponents) async

    final public func remove(_ workout: WorkoutPlan, at: DateComponents) async

    final public func markComplete(_ workout: WorkoutPlan, at: DateComponents) async

    final public func removeAllWorkouts() async
}
```

WorkoutKit

Planning

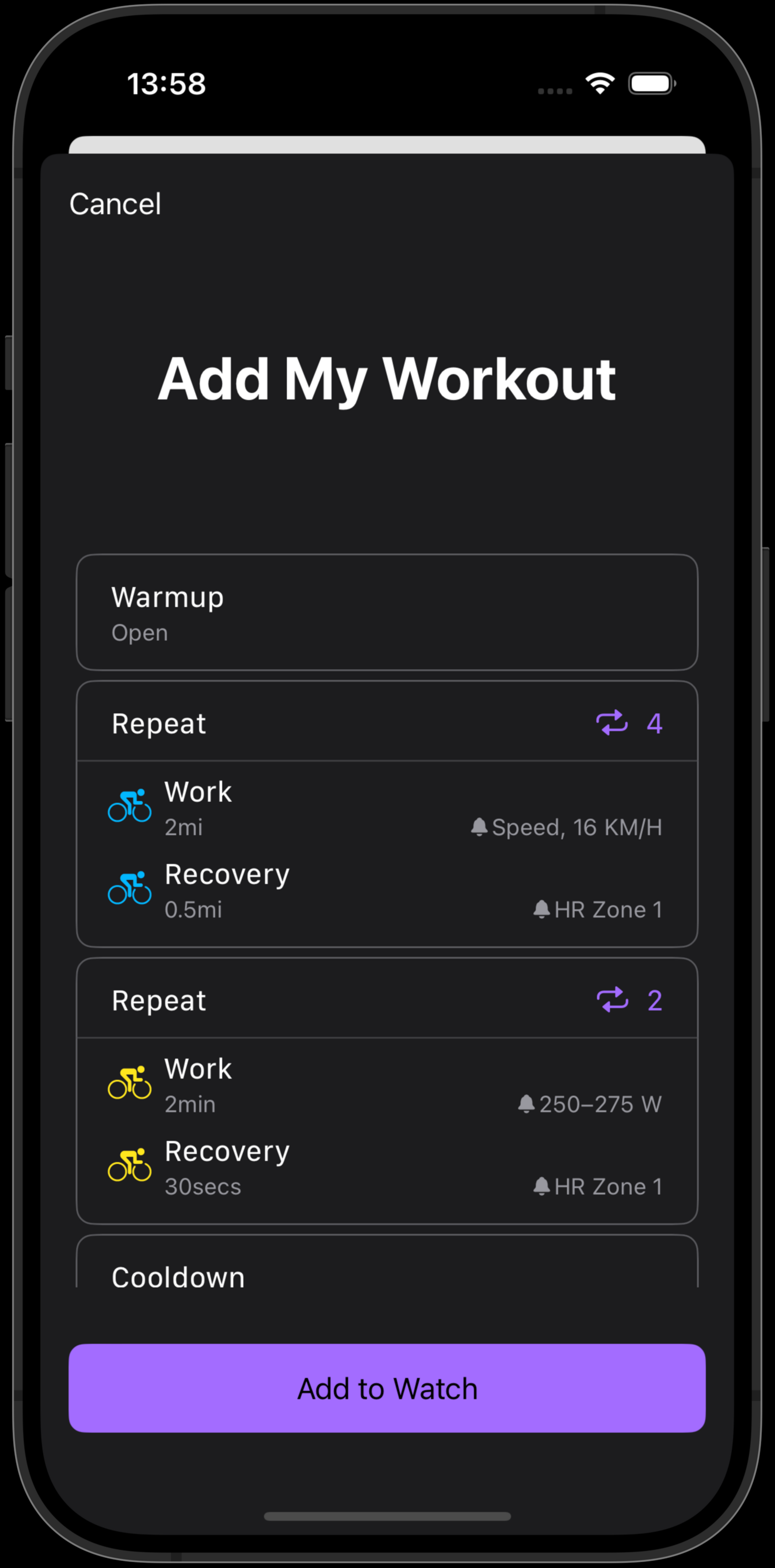
```swift
@available(iOS 17.0, watchOS 10.0, *)
extension HKWorkout {

    public var workoutPlan: WorkoutPlan? { get async throws }
}


@available(iOS 17.0, watchOS 10.0, *)
public struct WorkoutPlan : Equatable, Hashable, Sendable, Identifiable
```

WorkoutKit

Planning

```swift
public enum Workout : Equatable, Hashable, Sendable {

        case goal(SingleGoalWorkout)

        case custom(CustomWorkout)

        case pacer(PacerWorkout)

        case swimBikeRun(SwimBikeRunWorkout)
}
```

```
struct PresentPreviewDemo: View {
    private let cyclingWorkoutPlan: WorkoutPlan
    @State var showPreview: Bool = false

    init() {
        cyclingWorkoutPlan =
WorkoutPlan(.custom(WorkoutStore.createCyclingCustomWorkout()))
    }

    var body: some View {
        Button("Present Cycling Workout Preview") {
            showPreview.toggle()
        }
        .workoutPreview(cyclingWorkoutPlan, isPresented:
$showPreview)
    }
}
```



13:58

Cancel

Add My Workout

Warmup
Open

Repeat 4

Work
2mi Speed, 16 KM/H

Recovery
0.5mi HR Zone 1

Repeat 2

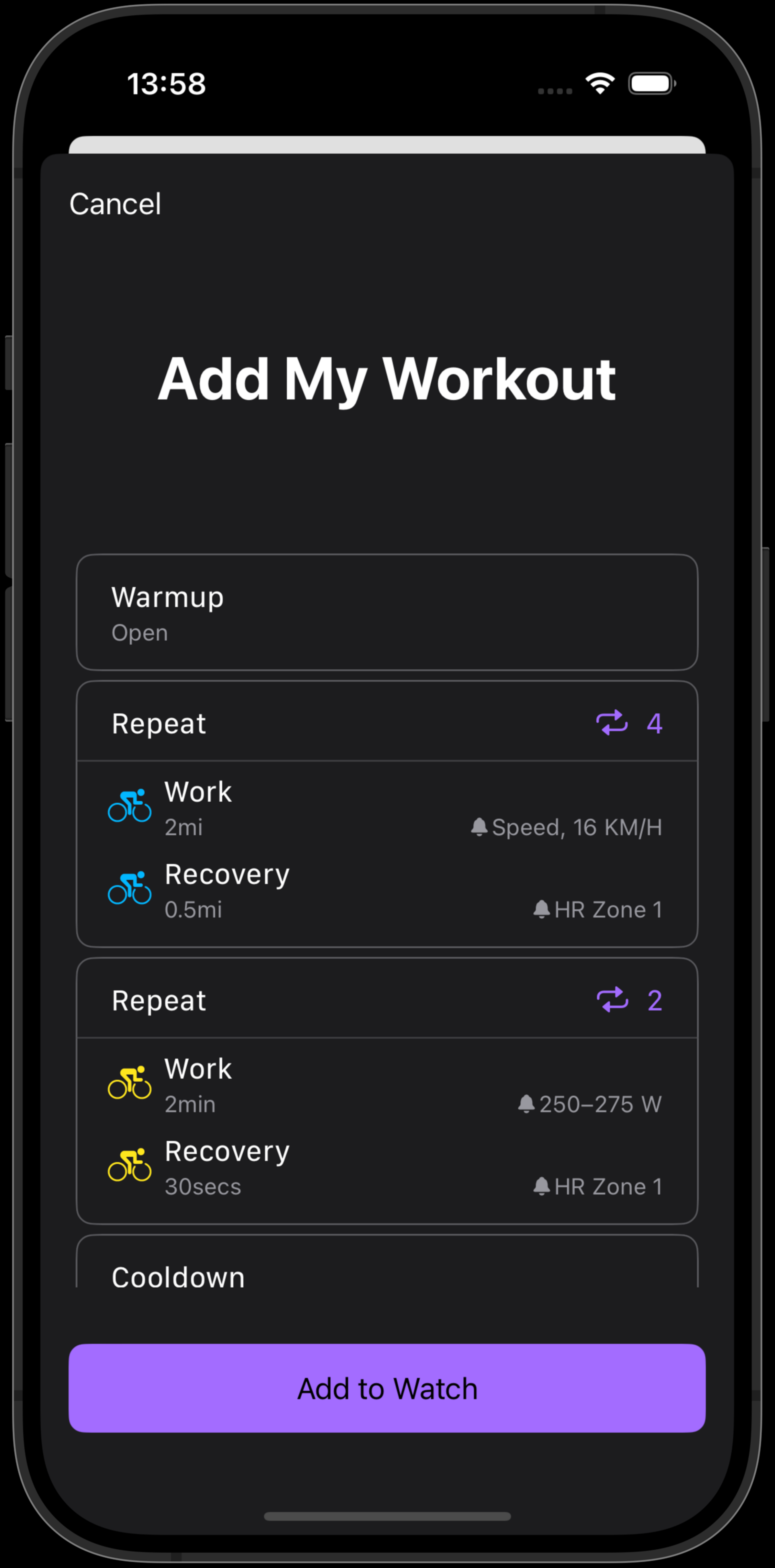
Work
2min 250–275 W

Recovery
30secs HR Zone 1

Cooldown

Add to Watch

```
struct PresentPreviewDemo: View {
    private let cyclingWorkoutPlan: WorkoutPlan
    @State var showPreview: Bool = false

    init() {
        cyclingWorkoutPlan =
WorkoutPlan(.custom(WorkoutStore.createCyclingCustomWorkout()))
    }

    var body: some View {
        Button("Present Cycling Workout Preview") {
            showPreview.toggle()
        }
        .workoutPreview(cyclingWorkoutPlan, isPresented:
$showPreview)
    }
}
```



13:58

Cancel

Add My Workout

Warmup
Open

Repeat 4

Work
2mi Speed, 16 KM/H

Recovery
0.5mi HR Zone 1

Repeat 2

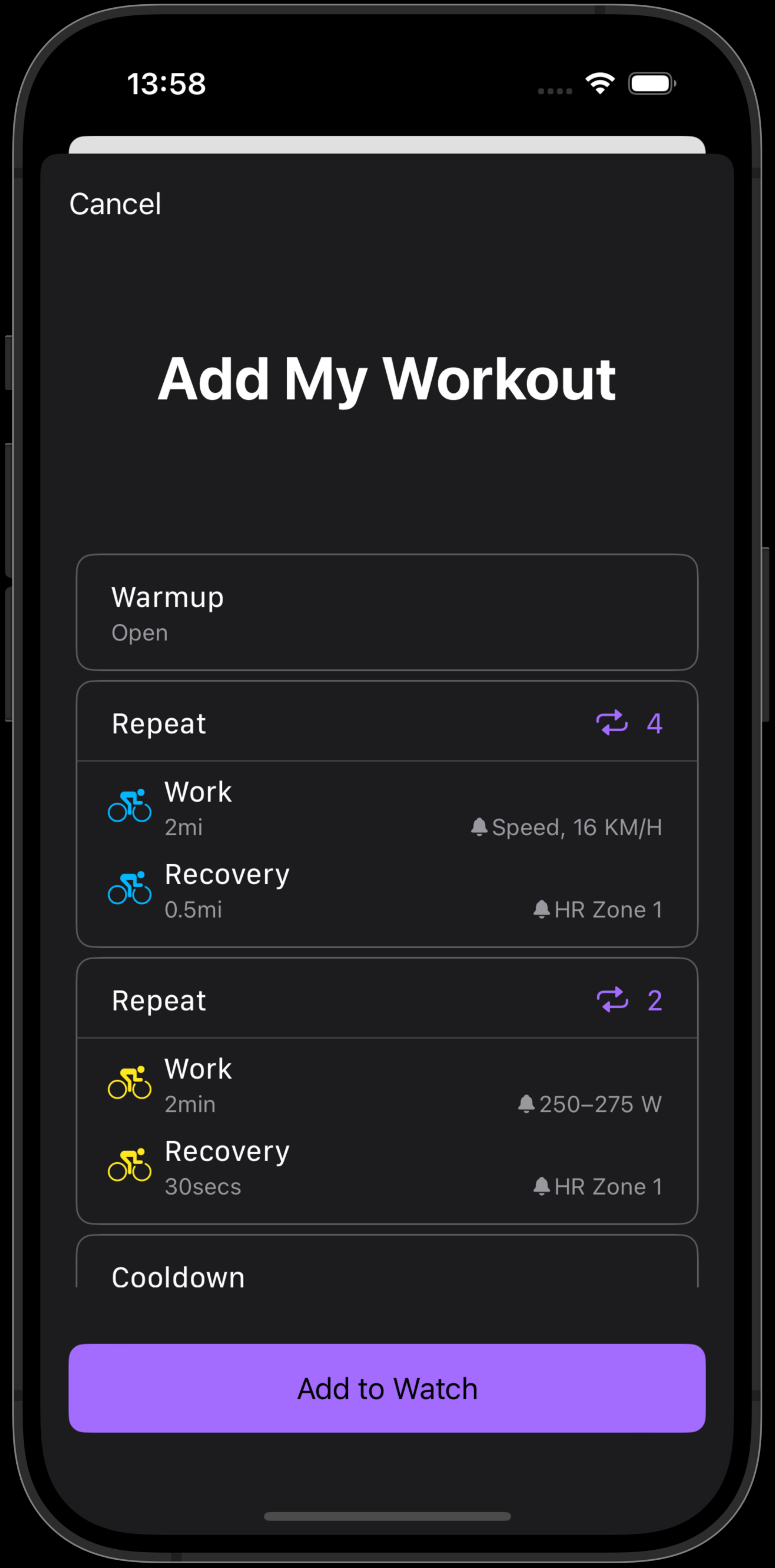
Work
2min 250–275 W

Recovery
30secs HR Zone 1

Cooldown

Add to Watch

```
struct PresentPreviewDemo: View {
    private let cyclingWorkoutPlan: WorkoutPlan
    @State var showPreview: Bool = false

    init() {
        cyclingWorkoutPlan =
WorkoutPlan(.custom(WorkoutStore.createCyclingCustomWorkout()))
    }

    var body: some View {
        Button("Present Cycling Workout Preview") {
            showPreview.toggle()
        }
        .workoutPreview(cyclingWorkoutPlan, isPresented:
$showPreview)
    }
}
```



Add My Workout

Warmup
Open

Repeat 4

Work
2mi
Speed, 16 KM/H

Recovery
0.5mi
HR Zone 1

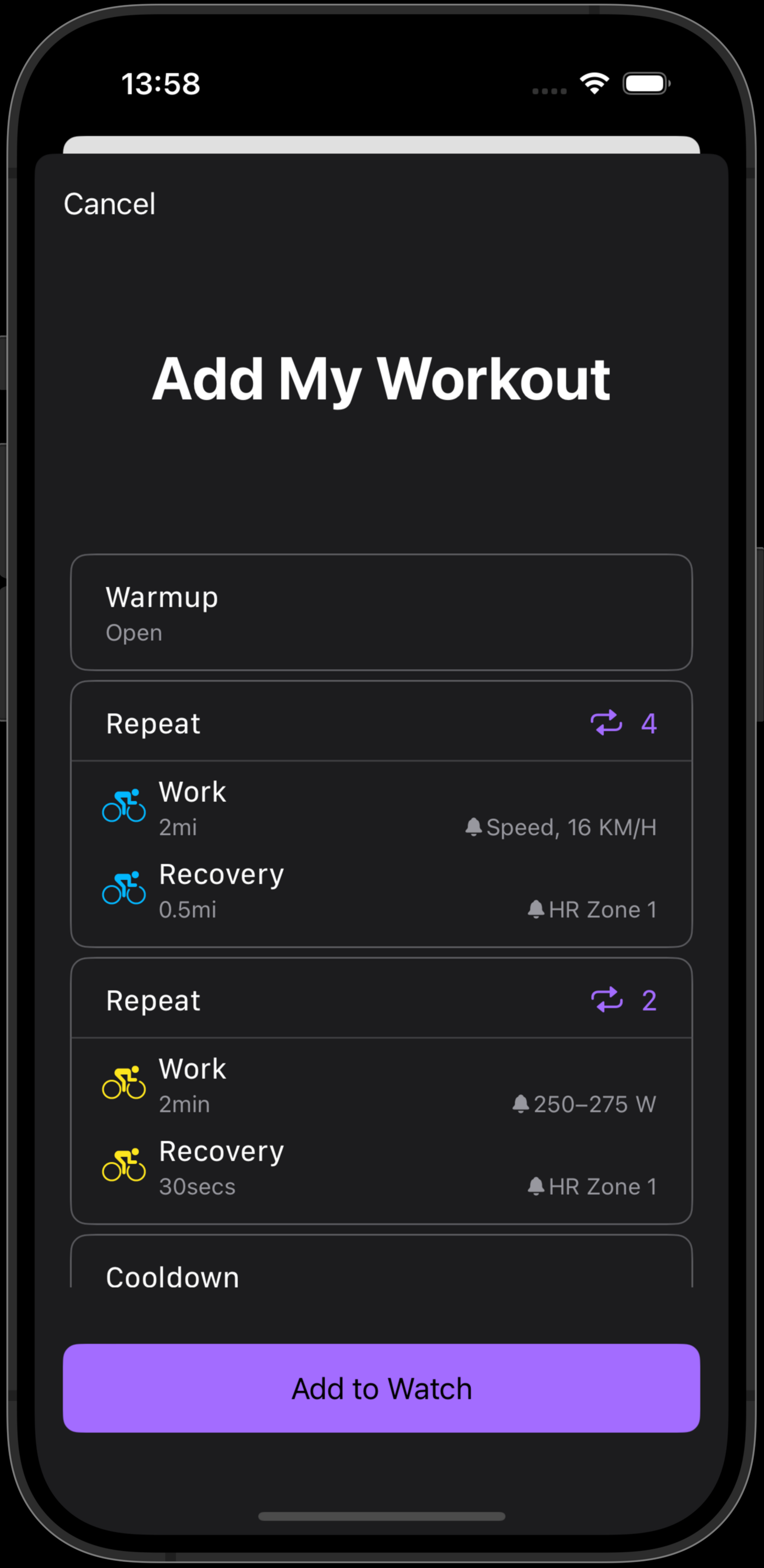
Repeat 2

Work
2min
250–275 W

Recovery
30secs
HR Zone 1

Cooldown

```
// Warmup step
let warmupStep = WorkoutStep()

// Block 1.
let block1 = Self.cyclingBlockOne()

// Block 2.
let block2 = Self.cyclingBlockTwo()

// Cooldown.
let cooldownStep = WorkoutStep(goal: .time(5, .minutes))

return CustomWorkout(activity: .cycling,
                     location: .outdoor,
                     displayName: "My Workout",
                     warmup: warmupStep,
                     blocks: [block1, block2],
                     cooldown: cooldownStep)
```



Add My Workout

Warmup
Open

Repeat 4

Work
2mi
Speed, 16 KM/H

Recovery
0.5mi
HR Zone 1

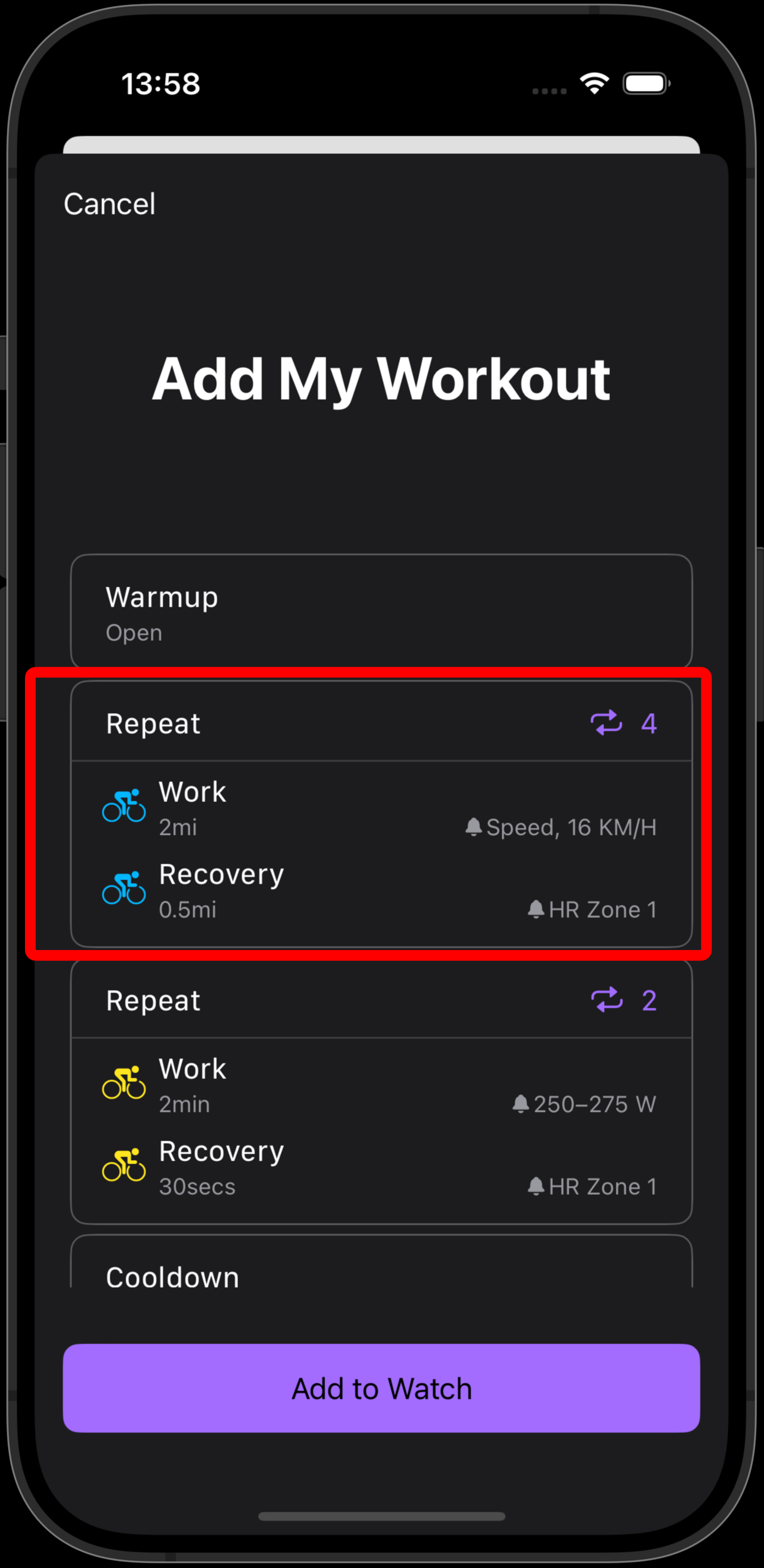
Repeat 2

Work
2min
250–275 W

Recovery
30secs
HR Zone 1

Cooldown

```swift
// Work step 1.
var workStep1 = IntervalStep(.work)
workStep1.step.goal = .distance(2, .miles)
workStep1.step.alert = .speed(10, unit: .milesPerHour, metric: .current)

// Recovery step.
var recoveryStep1 = IntervalStep(.recovery)
recoveryStep1.step.goal = .distance(0.5, .miles)
recoveryStep1.step.alert = .heartRate(zone: 1)

return IntervalBlock(steps: [workStep1, recoveryStep1],
                     iterations: 4)
```



13:58

Cancel

Add My Workout

Warmup
Open

Repeat 4

Work
2mi — Speed, 16 KM/H

Recovery
0.5mi — HR Zone 1

Repeat 2

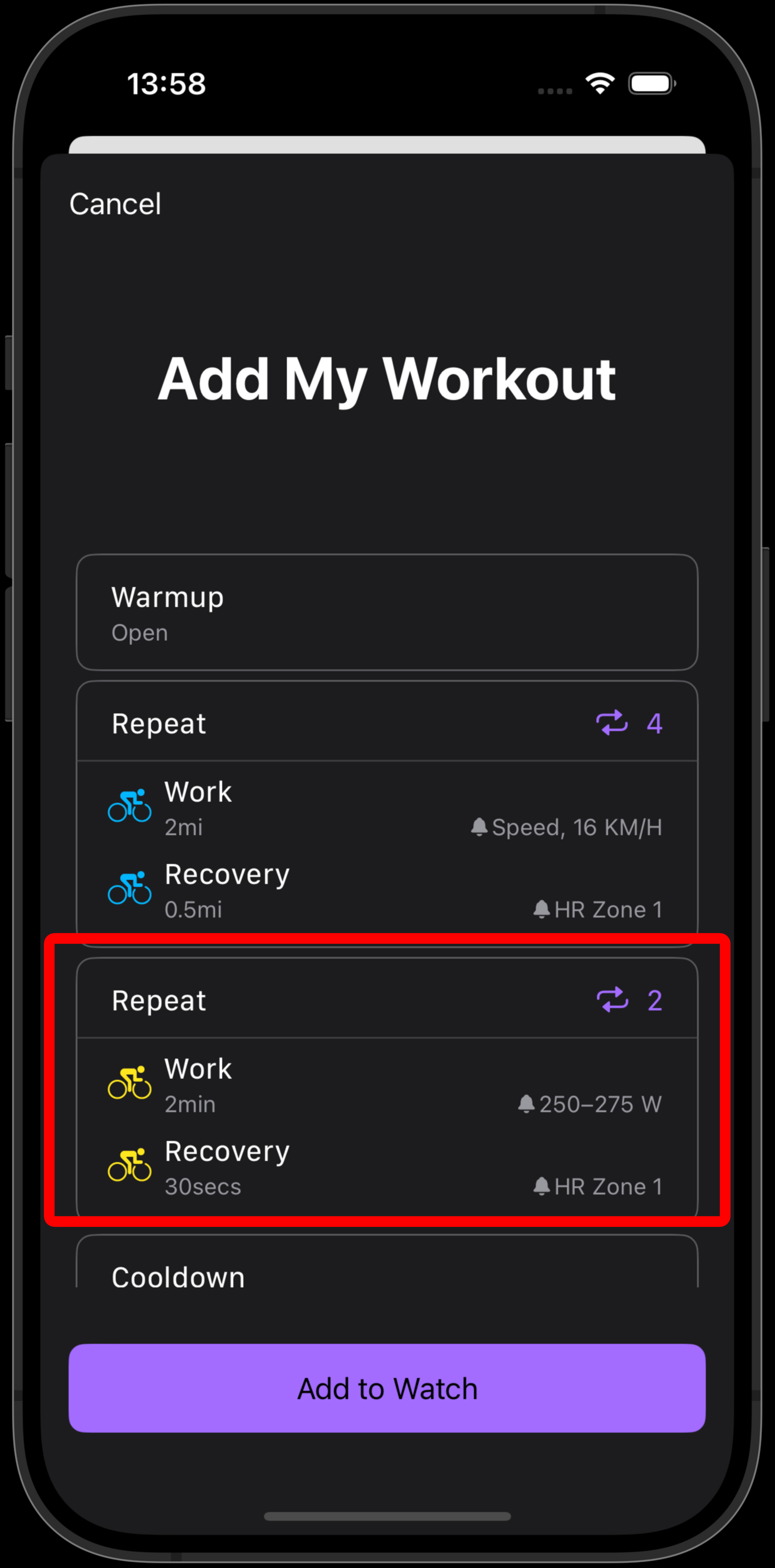
Work
2min — 250–275 W

Recovery
30secs — HR Zone 1

Cooldown

Add to Watch

```swift
// Work step.
var workStep2 = IntervalStep(.work)
workStep2.step.goal = .time(2, .minutes)
workStep2.step.alert = .power(250...275, unit: .watts)

// Recovery step.
var recoveryStep2 = IntervalStep(.recovery)
recoveryStep2.step.goal = .time(30, .seconds)
recoveryStep2.step.alert = .heartRate(zone: 1)

// Block with two iterations.
return IntervalBlock(steps: [workStep2, recoveryStep2],
                     iterations: 2)
```



13:58

Cancel

Add My Workout

Warmup
Open

Repeat 4

Work
2mi
Speed, 16 KM/H

Recovery
0.5mi
HR Zone 1

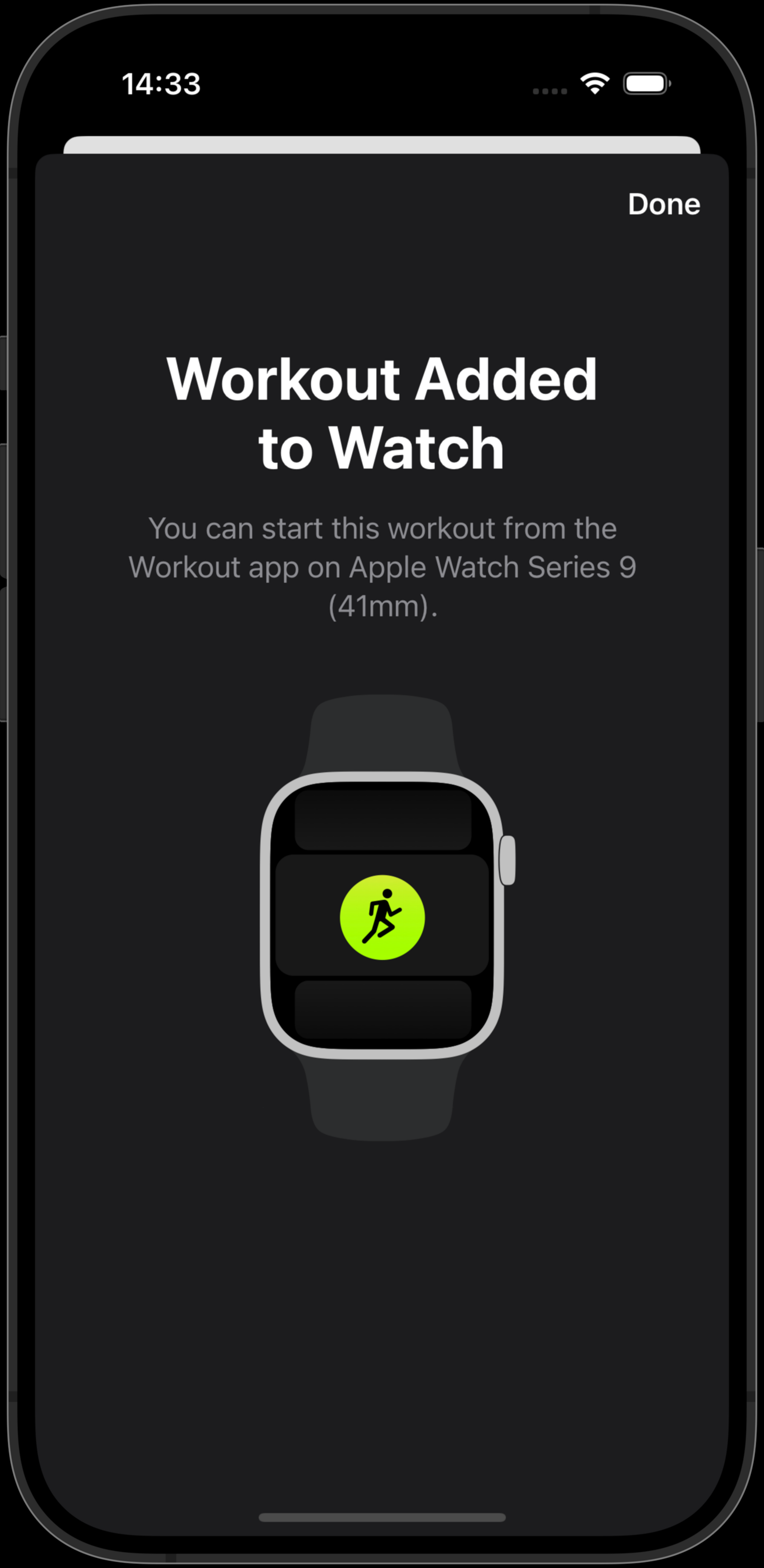
Repeat 2

Work
2min
250–275 W

Recovery
30secs
HR Zone 1

Cooldown

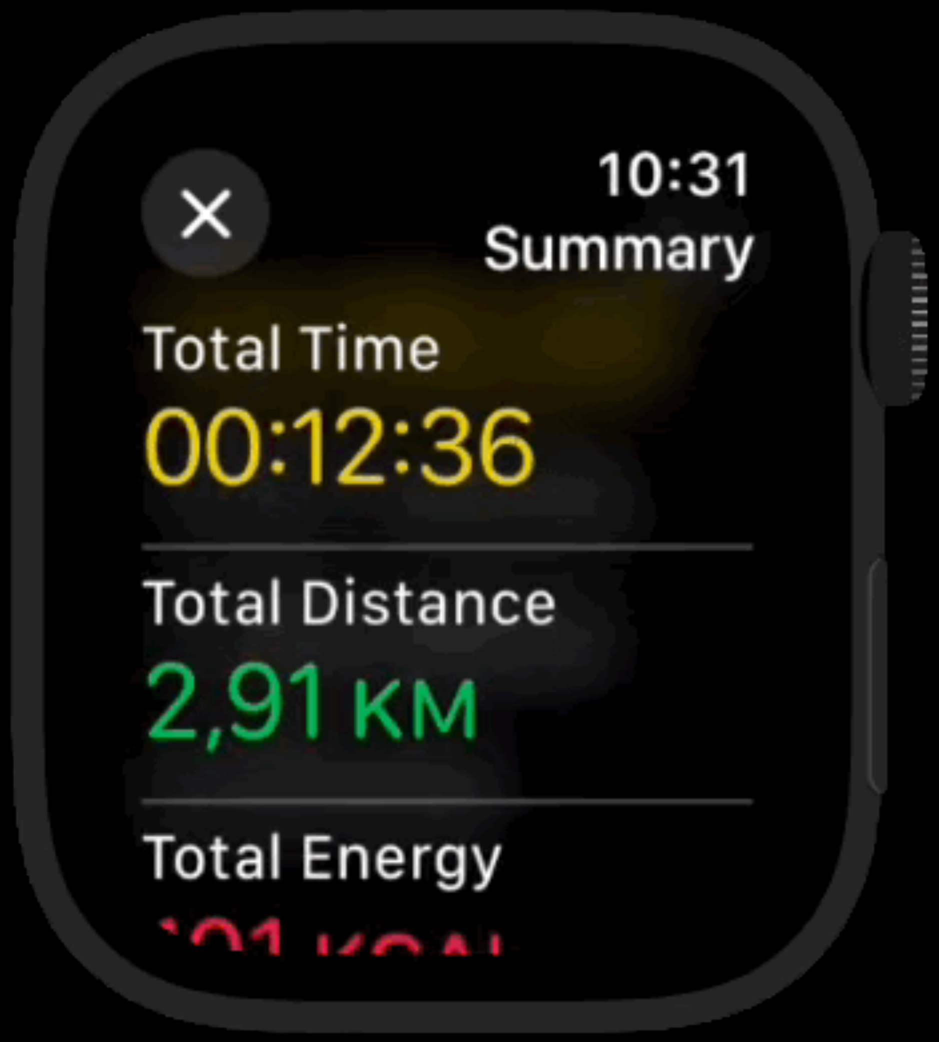
Add to Watch



Workout Added to Watch

You can start this workout from the Workout app on Apple Watch Series 9 (41mm).

Metrics



10:31

Summary

Total Time

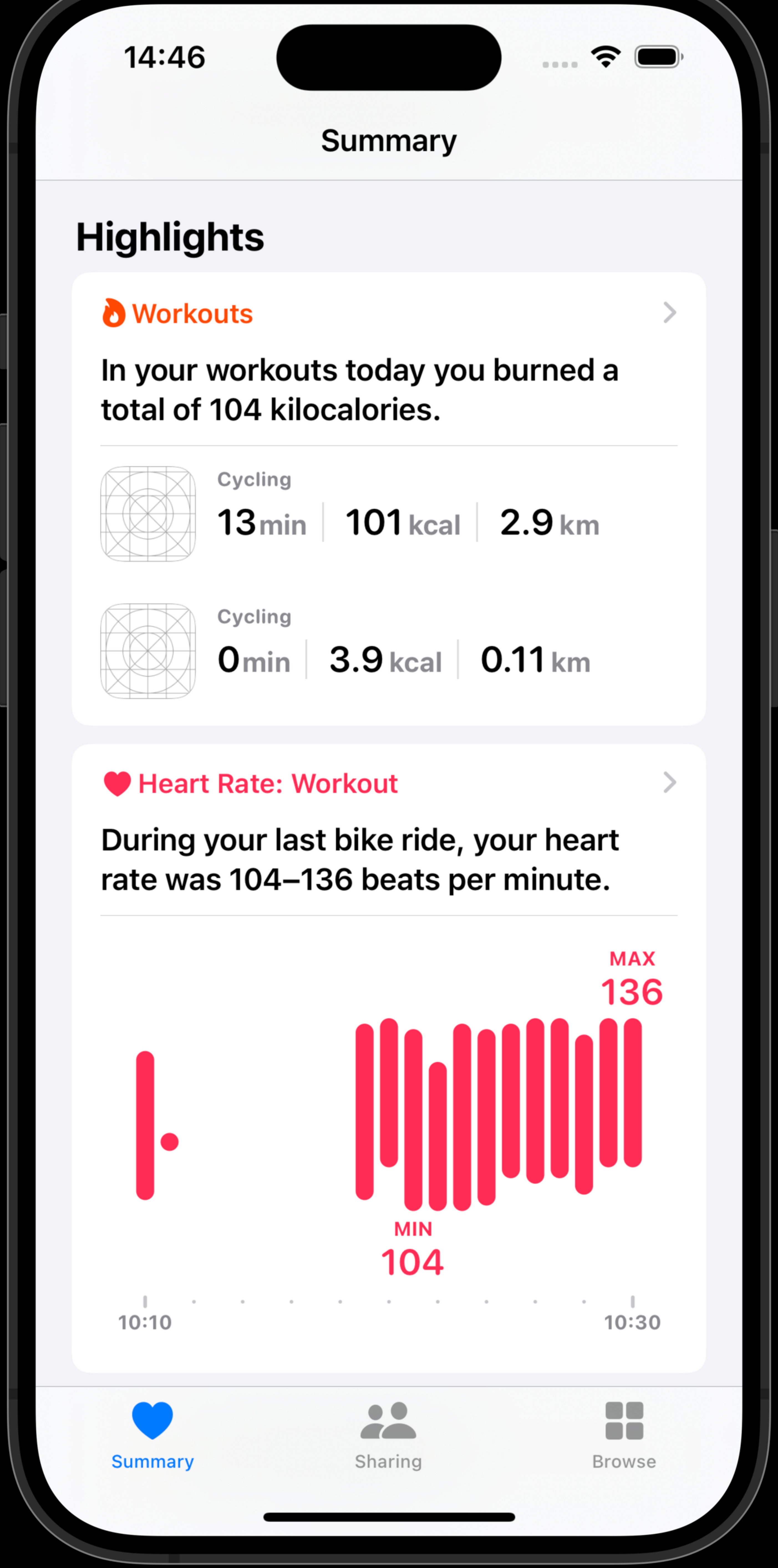
00:12:36

Total Distance

2,91 KM

Total Energy

101 KCAL



Highlights

Workouts

In your workouts today you burned a total of 104 kilocalories.

Cycling
13 min | 101 kcal | 2.9 km

Cycling
0 min | 3.9 kcal | 0.11 km

Heart Rate: Workout

During your last bike ride, your heart rate was 104–136 beats per minute.

10 + Sensors

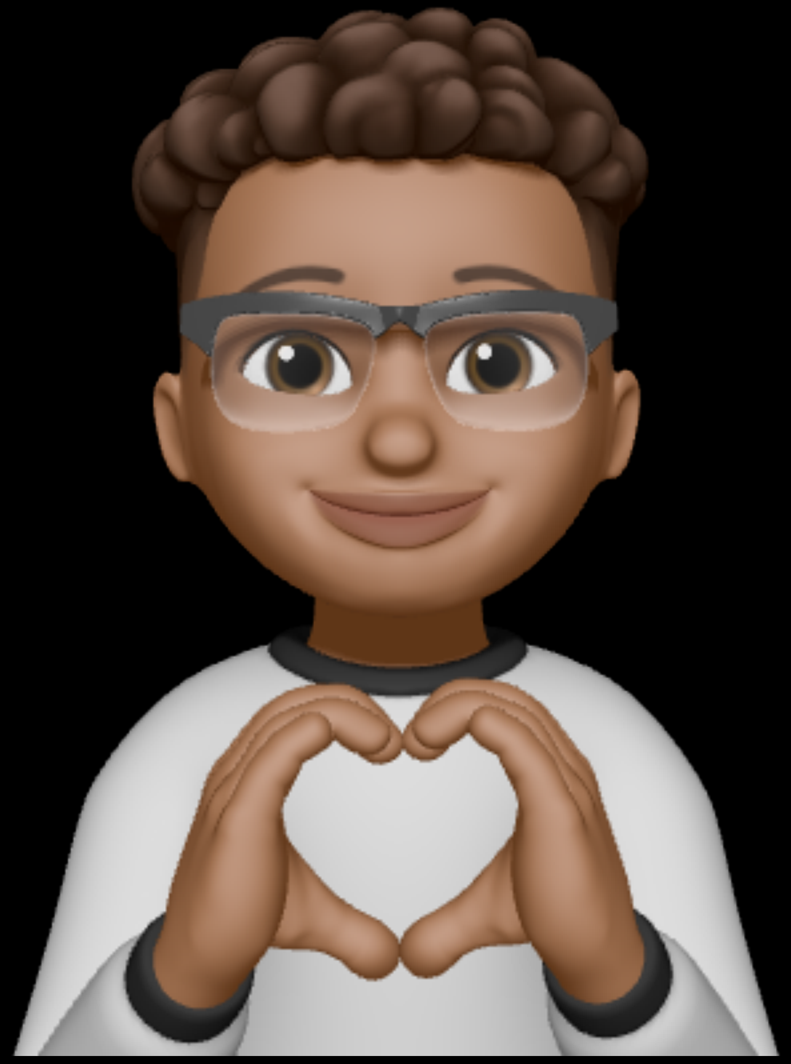
wahoo
SPEED
wahoo
CADENCE

Skandi
wahoo
CADENCE
COOSPO

BROMPTON

wahoo
CADENCE
SHIMANO
ULT

Grazie!



Slides

Proton Drive

The safest way to store or share your files

Proton Drive

drive.proton.me