

Claude Initial Version (using 3.7)

SF Street Cleaning Parking Assistant: App Development Plan

I'll create a comprehensive plan for your parking ticket avoidance app. This is an excellent idea that solves a real problem many San Francisco residents face!

App Design & Features

Core Functionality

- Map-based interface showing car location(s) and street cleaning zones
- Countdown timers for upcoming street cleaning events
- Multi-user alerts and notifications
- AirTag integration for real-time car location tracking
- Street cleaning schedule data integration

User Interface Design

1. **Home Screen:** Map view showing:
 - Car location(s) with pins
 - Color-coded streets showing cleaning schedules
 - Countdown to next cleaning event
2. **Car Management Screen:**
 - Add/remove cars
 - Link AirTags to specific vehicles
 - Set manual locations (backup for AirTag)
3. **Alert Settings:**
 - Configure notification timing (1hr before, night before)
 - Set notification methods (push, SMS)
 - Family member sharing management

Technical Architecture

Frontend (iOS App)

- **Language/Framework:** Swift with SwiftUI for modern UI components
- **Key Libraries:**
 - MapKit for mapping functionality
 - CoreLocation for device positioning
 - UserNotifications for push notifications
 - FindMy framework for AirTag integration

Backend Services

- **Server:** Node.js or Python-based service
- **Database:** MongoDB for flexibility with location data
- **API Layer:** RESTful API for client-server communication
- **Cloud Provider:** AWS or Firebase (easier for beginners)

Data Sources & Integration

1. **SF Street Cleaning Data:**
 - SF Open Data Portal: <https://data.sfgov.org/City-Infrastructure/Street-Sweeping-Schedule/yhqp-rigs>
 - Will need regular syncing to keep schedules updated
2. **AirTag Integration:**
 - Use Apple's FindMy framework (requires Apple Developer account)
 - Fallback: Manual location setting if AirTag data unavailable
3. **Notification System:**
 - Apple Push Notification Service (APNS)
 - Twilio for SMS capabilities

Development Roadmap

Phase 1: Setup & Core Infrastructure

1. Create iOS project structure
2. Set up backend server and database
3. Implement authentication system
4. Create data models for vehicles, users, locations

Phase 2: Map & Location Features

1. Implement MapKit integration
2. Build street cleaning data parser
3. Create location tracking system
4. Implement geofencing for location alerts

Phase 3: Notification & Sharing Systems

1. Build notification scheduling system
2. Implement multi-user sharing
3. Connect AirTag integration
4. Create alert rules engine

Phase 4: Testing & Refinement

1. User testing with SF residents
2. Performance optimization
3. Edge case handling (data gaps, connection issues)
4. Battery usage optimization

Code Structure & Implementation

iOS App (Swift)

```
// Example structure for main view controller
import UIKit
import MapKit
import CoreLocation

class MainViewController: UIViewController,
CLLocationManagerDelegate, MKMapViewDelegate {

    @IBOutlet weak var mapView: MKMapView!
    private let locationManager = CLLocationManager()

    override func viewDidLoad() {
        super.viewDidLoad()
        setupLocationManager()
        setupMapView()
        loadVehicleLocations()
        checkUpcomingStreetCleaning()
    }

    // Location manager setup
    private func setupLocationManager() {
        locationManager.delegate = self
        locationManager.desiredAccuracy =
kCLLocationAccuracyBest
        locationManager.requestWhenInUseAuthorization()
        locationManager.startUpdatingLocation()
    }
}
```

```

    }

    // Map configuration
    private func setupMapView() {
        mapView.delegate = self
        mapView.showsUserLocation = true
        // Additional map configuration
    }

    // Load vehicle data from backend
    private func loadVehicleLocations() {
        // API call to get stored vehicle locations
    }

    // Check for upcoming street cleaning events
    private func checkUpcomingStreetCleaning() {
        // Query for nearby street cleaning schedules
        // Schedule notifications if needed
    }
}

```

Backend API (Node.js)

```

// Example Express route for checking street cleaning
const express = require('express');
const router = express.Router();
const StreetCleaningService =
require('../services/streetCleaningService');

router.post('/check-street-cleaning', async (req, res) => {
  try {
    const { latitude, longitude } = req.body;
    const service = new StreetCleaningService();

    // Find upcoming street cleaning events near location
    const events = await service.findNearbyEvents(latitude,
longitude);

    // Return events to client
    res.json({ success: true, events });
  } catch (error) {
    console.error(error);
    res.status(500).json({ success: false, message: 'Server
error' });
  }
}

```

```
});  
  
module.exports = router;
```

Data Storage Strategy

User & Vehicle Data

- MongoDB collections for:
 - Users (account info, preferences)
 - Vehicles (make, model, AirTag ID)
 - Locations (historical parking spots)
 - Alerts (notification history)

Street Cleaning Data

- Initially load from SF Open Data API
- Store processed data in MongoDB
- Update weekly with scheduled job
- Create spatial indices for fast location-based queries

Launch Strategy

1. **Development Environment:**
 - Xcode for iOS development
 - GitHub for version control
 - TestFlight for beta testing
2. **Testing Phases:**
 - Alpha: Internal testing with simulated data
 - Beta: Limited user group in SF with real data
 - Public launch: App Store release
3. **App Store Preparation:**
 - Create compelling screenshots and App Store description
 - Prepare privacy policy (important for location data)
 - Set up App Store Connect account

Potential Challenges & Solutions

1. **AirTag Integration Limitations:**
 - Challenge: Apple's FindMy API has restrictions
 - Solution: Implement manual location fallback
2. **Battery Usage:**

- Challenge: Location tracking can drain battery
- Solution: Smart polling based on proximity to cleaning time
- 3. **Data Accuracy:**
 - Challenge: SF street cleaning data may have gaps
 - Solution: Allow user corrections and community reporting
- 4. **Privacy Concerns:**
 - Challenge: Managing location data securely
 - Solution: Implement proper encryption and data retention policies

Would you like me to elaborate on any specific part of this plan? Or should we discuss the next steps to begin implementation?