Requirements Specification

for

SharkPark

Bighani Lometillo
Charles Milton
Ly Nguyen
Zachary Padilla
Lawrence Degoma

Version 1.0
September 23, 2025

Table of Contents

Executive Summary

SharkPark is a mobile application designed to address one of the most common complaints faced by CSULB students and staff, the lack of parking. With approximately 40,000 students competing for 14,000 spaces, demand consistently exceeds supply during peak hours, resulting in wasted time, increased frustration, and negative impacts on both academic performance and the environment. SharkPark provides a modern, data-driven approach to this longstanding problem by transforming students' mobile devices into privacy preserving sensors that contribute real time parking occupancy data.

The system works by leveraging geofencing and activity recognition on student phones to capture anonymous "enter" and "leave" events as vehicles arrive and depart. These signals are aggregated by a backend service that applies machine learning models to estimate current lot occupancy and generate short and long term forecasts. Students using the app benefit from up to date parking availability, personalized lot recommendation based on location and predicted demand, and fallback options if their lot of choice is near or at capacity.

Beyond individual convenience, SharkPark supports broader institutional goals. Campus Parking Services and university administration gain valuable insights into usage trends, peak demand times, and the impact of special events, enabling data-informed planning and policy decisions. Integration with shuttle services and third-party data further enhances forecasting accuracy and provides students with multimodal commuting options.

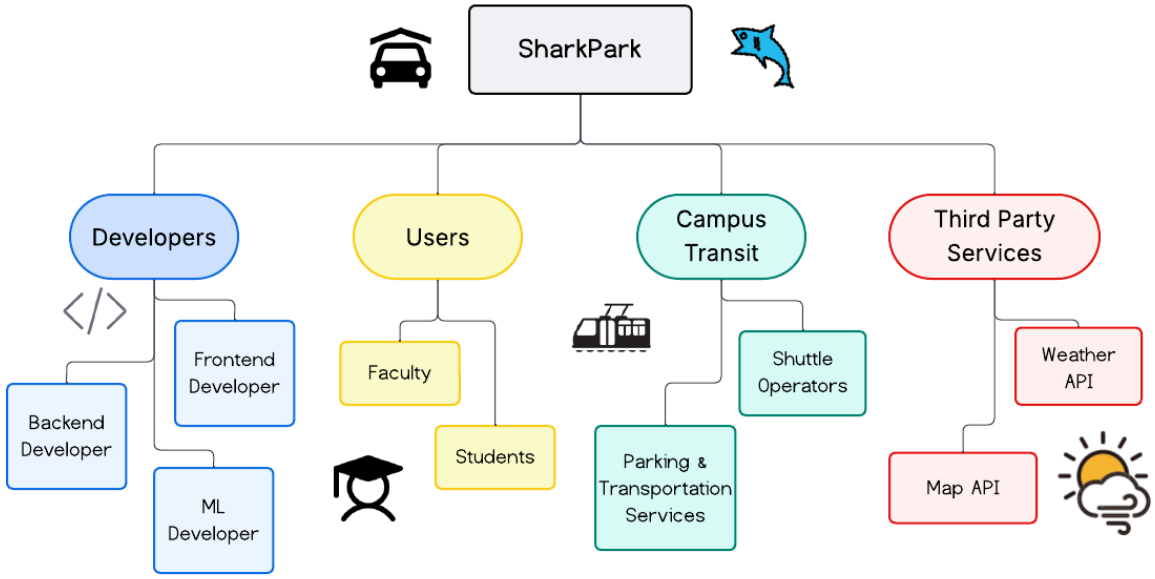
Key design principles include privacy, efficiency, and scalability. The app is designed to minimize battery consumption by batching events and using low power sensors, while ensuring that no personally identifiable information or raw GPS tracks are stored. The backend architecture is modular and scalable, capable of handling thousands of concurrent users and spikes in demand around class start times. By prioritizing lightweight and cost effective mobile sensing over infrastructure heavy solutions such as cameras or fixed sensors, SharkPark ensures long term feasibility and adaptability.

Ultimately, SharkPark aims to reduce the average time spent searching for parking, lower commuter stress, and contribute to sustainability by cutting down on unnecessary driving. For students, this means fewer late arrivals and a smoother start to their days. For the university, it means improved operational efficiency and a better commuter experience that enhances the overall perception of campus life. By bridging real time technology with practical commuter needs, SharkPark offers a forward looking solution to one of CSULB's most persistent challenges.

Stakeholder Model

Stakeholders represent all parties that contribute to or are impacted by SharkPark. Below we define the primary stakeholders, their roles, responsibilities, and expected deliverables. Primary stakeholders include Students and faculty (app users), Developers and Devops (build and maintain systems), Campus Transit (shuttle information), and Third-Party Services (weather, maps). Each stakeholder group has specific expectations: app users provide anonymous signals and feedback, parking services may supply lot metadata and event schedules, and the development team ensures system reliability and data privacy.

Stakeholder Diagram



Stakeholders

Developers

Representatives	Bighani Lometillo, Charles Milton, Ly Nguyen, Zachary Padilla, Lawrence Degoma
Description	Build the software and systems for the app
Type(s)	Frontend developer, backend developer, project manager, machine learning developer
Responsibilities	Implement app and backend, monitor performance
Success Criteria	Completed development cycle and deployment
Involvement	Full development cycles
Deliverables	App releases, updates, and monitoring

Users

Representatives	Students and faculty
Description	Provide anonymous enter/leave signals
Responsibilities	Use the app
Deliverables	Signal data, feedback

Campus Transit

Representatives	Shuttle operators, Parking & Transportation Services
Description	Integrations and operations
Responsibilities	Provide real time shuttle locations and ETAs
Deliverables	Shuttle feed

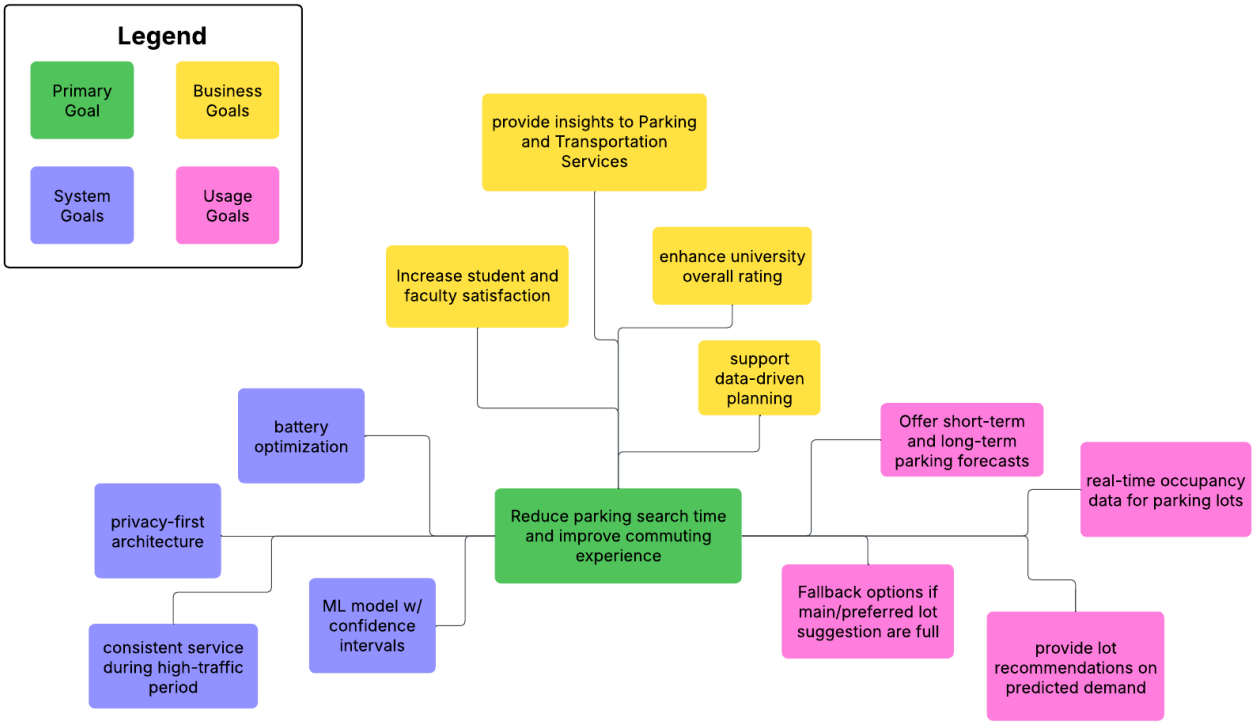
Third Party Services

Representatives	Weather/maps API
Description	Provide external data
Responsibilities	API integration
Deliverables	APIs and docs

Goal Model

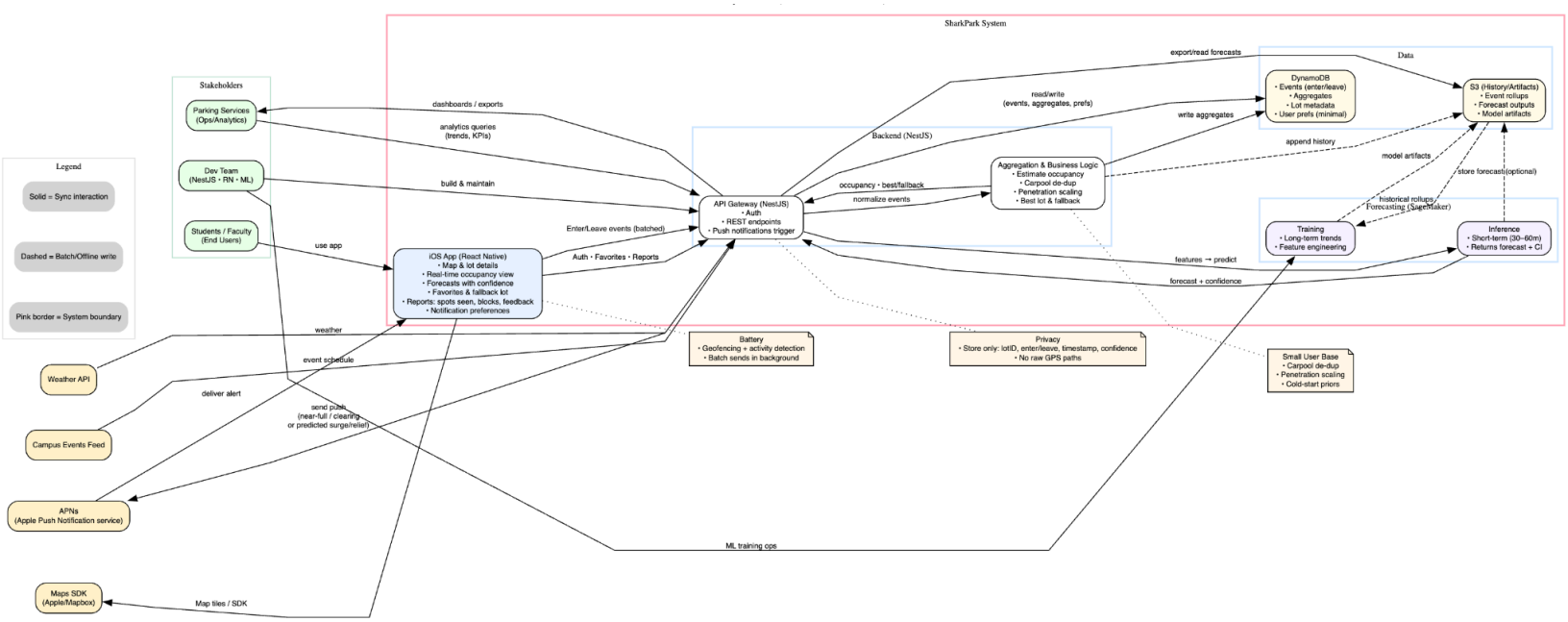
The primary goal of SharkPark is to reduce the amount of time students spend searching for parking and to improve the reliability of student commuting. This primary goal is supported by Business Goals to increase satisfaction and provide campus insights, Usage Goals for accurate real time occupancy and clear recommendations, and System goals towards privacy first, reliability, and scalability. The diagram below visualizes how the primary goal is broken up into these categories.

1. **Primary Goals**: These goals are designed to reduce the time students and faculty spend on parking and improve overall commuting experience. They provide real-time information and predictive analytics to enable users to plan ahead.
2. **Business Goals**: These goals seek to minimize frustration and improve overall satisfaction with their parking experience. They deliver data analytics to aid CSULB's Parking and Transportation Services in decision making, lot management, and event planning.
3. **Usage Goals**: These goals provide users with real-time analytics, aggregated occupancy, and suggested lot displayed with status indicators. They offer personalized recommendations based on forecasted demand and estimated availability to improve parking efficiency.
4. **System Goals**: These goals ensure user privacy through the anonymous collection of entry and exit events without personal identifiers or history. They are focused on scalability, reliability, and energy efficiency through optimized background processing.



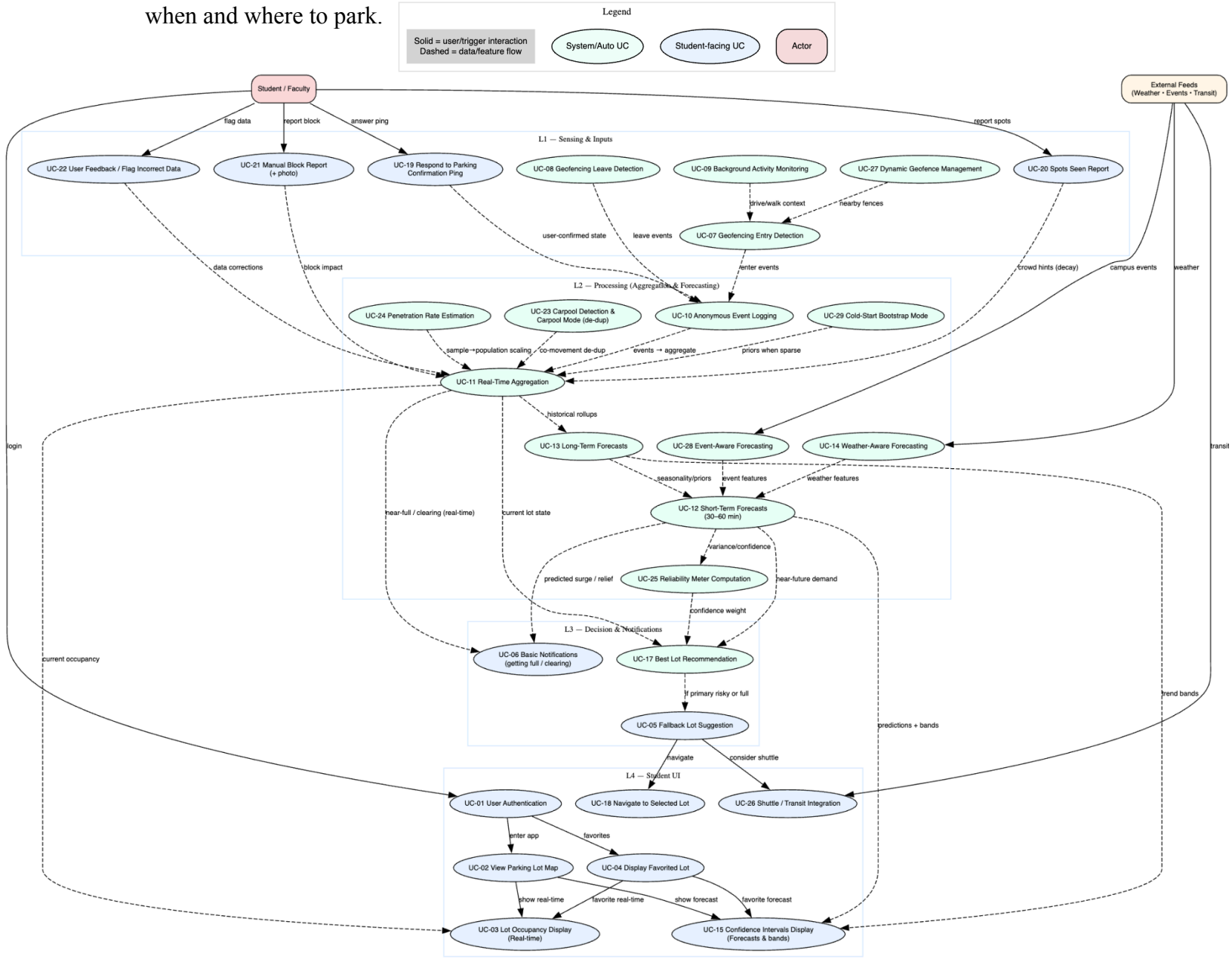
System Vision

SharkPark's vision is a privacy preserving, low cost parking guidance system that uses mobile phones as anonymous sensors. Student devices detect entry/exit geofenced lots and send minimal events to the backend. The backend aggregates events, estimates lot occupancy, runs short and long term forecasting models, and exposes APIs for the mobile client dashboard. The system will support offline caching, battery optimized background processing, and careful degradation when data is sparse.



Usage Model

The usage model for SharkPark describes how students, faculty and the system interact to reduce parking frustration and improve the commuting experience. Students and faculty (end users) open the SharkPark mobile app to view real-time parking availability across campus. They can log in, see lot occupancy, check forecasted availability, and receive notifications when their preferred lots are filling up or clearing out. Users can also favorite specific lots, report open spots or blocked areas, and view fallback recommendations when their favorite lot is full. The backend system aggregates anonymous “enter” and “leave” signals detected through geofencing, calculates real-time occupancy, and generates short- and long-term forecasts using machine learning. Forecasts and occupancy data are displayed within the app, which will help users plan when and where to park.



Use Cases

1. User Authentication

Use Case #1	User Authentication	
Goal in Context	Allows students to sign in using any email	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	● User has valid credentials ● User has network access	
Success End Condition	The user has created an account and is logged in	
Failed End Condition	Invalid credentials, retry option	
Primary, Secondary Actors	**Primary Actor:** Student/Faculty **Secondary Actor:** System	
Trigger	User opens the app and presses “login” button	
Description	**Step**	**Action**
	1.	System prompts user to enter “login information” or “create account”
	2.	User chooses an account
	3.	User enters cell number
	4.	The system validates the number
	5.	System sends a confirmation code to entered number
Extensions	**Step**	**Branching Action**
	1a	The number returns as invalid, retry
Sub-variation	**Step**	**Branching Action**

	1	System shows error, network failure, retry
Related Information		
Priority	High	
Performance	1-5 minutes	
Frequency	Daily	
Channels to actors	Database	
OPEN ISSUES	N/A	
Due Date	December 2025	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

2. View Parking Lot Map

Use Case #2	View Parking Lot Map	
Goal in Context	Display interactive campus parking map with lot statuses	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	User is authenticated	
Success End Condition	Map displays lots with color coded occupancy	
Failed End Condition	Map fails to load	
Primary, Secondary Actors	**Primary:** User (Students and faculty)	
Trigger	User opens ‘Map’	
Description	**Step**	**Action**
	1.	App requests lot data
	2.	Backend returns lot data from database
	3.	Map displays lots with color coded occupancy
Extensions	**Step**	**Branching Action**
	1a	Map fails to load, retry
Sub-variation	**Step**	**Branching Action**
	1	Map fails to load, display cached map
Related Information		
Priority	High	
Performance	<1 minute	
Frequency	Multiple times daily	
Channels to actors	Interactive via mobile interface	

OPEN ISSUES	N/A
Due Date	December 2025
…any other management info	N/A
Superordinates	N/A
Subordinates	N/A

3. Lot Occupancy Display

Use Case #3	Lot Occupancy Display	
Goal in Context	Show percentage fullness and status for each lot	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	Map is loaded	
Success End Condition	System displays percentage fullness and occupancy category (Low/Med/High)	
Failed End Condition	System fails to load	
Primary, Secondary Actors	**Primary:** User (Students and faculty)	
Trigger	User taps a lot or views map legend	
Description	**Step**	**Action**
	1.	System displays percentage fullness and occupancy category
Extensions	**Step**	**Branching Action**
	1a	Stale data → show confidence rating

Sub-variation	Step	Branching Action
	1	Fail to load, refresh
Related Information		
Priority	High	
Performance	<1 minute	
Frequency	Multiple times daily	
Channels to actors	Database	
OPEN ISSUES	N/A	
Due Date	December 2025	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

4. Favorited Lot

Use Case #4	Display Favorited Lot
Goal in Context	Display lot occupancy for user's favorite lot
Scope & Level	**Scope:** Sharkpark **Level:** Primary
Preconditions	User location is enabled
Success End Condition	UI displays route and occupancy
Failed End Condition	No location found, low confidence
Primary, Secondary Actors	**Primary:** Users (Students and faculty)

Trigger	User opens ‘Favorite(s)’	
Description	**Step**	**Action**
	1.	System evaluates selected lot(s)
	2.	Forecasts and returns favorited forecast
	3.	UI displays route and estimated availability
Extensions	**Step**	**Branching Action**
	1a	No location Ask permission
Sub-variation	**Step**	**Branching Action**
	1	Sparse Data Show low confidence
Related Information		
Priority	High	
Performance	<1 minute	
Frequency	Semi-often	
Channels to actors	Interactive UI	
OPEN ISSUES	N/A	
Due Date	December 2025	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

5. Fallback Lot

Use Case #5	Fallback Lot Suggestion	
Goal in Context	Suggest backup lots if primary fills	
Scope & Level	**Scope:** Sharkpark **Level:** Primary	
Preconditions	Favorited lot returned	
Success End Condition	Displays fallback option	
Failed End Condition	All lots full	
Primary, Secondary Actors	**Primary:** Users (Students and faculty)	
Trigger	User views favorited lot (full/almost)	
Description	**Step**	**Action**
	1.	System computes fallback based on proximity and confidence
	2.	Display fallback option
Extensions	**Step**	**Branching Action**
	1a	All lots full Advise different arrival time
Sub-variation	**Step**	**Branching Action**
	1	All lots full Advise public transit
Related Information		
Priority	High	
Performance	<1 minute	
Frequency	Semi-often	
Channels to actors	Backend/Database	

OPEN ISSUES	N/A
Due Date	December 2025
…any other management info	N/A
Superordinates	N/A
Subordinates	N/A

6. Basic Notifications

Use Case #6	Basic Notifications	
Goal in Context	Push alerts for lot capacity changes	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	User has notifications enabled	
Success End Condition	User receives alert	
Failed End Condition	User has notifications disabled	
Primary, Secondary Actors	**Primary:** User (Students and faculty)	
Trigger	System detects lot near capacity or clearing	
Description	**Step**	**Action**
	1.	System sends push per user settings
	2.	User receives alert
Extensions	**Step**	**Branching Action**
	1a	User disables notification

Sub-variation	Step	Branching Action
	1	Notification fails to send
Related Information		
Priority	Medium	
Performance	<30 seconds	
Frequency	Periodic	
Channels to actors	Database	
OPEN ISSUES	N/A	
Due Date	December 2025	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

7. Geofencing Entry Detection

Use Case #7	Geofencing Entry Detection
Goal in Context	Detect when phone enters lot boundary
Scope & Level	**Scope:** SharkPark **Level:** Primary
Preconditions	App installed with geofences active
Success End Condition	Sends anonymized event to backend when possible
Failed End Condition	Data fails to send to backend
Primary, Secondary Actors	**Primary:** Users (Students and faculty)

Trigger	Device crosses geofence	
Description	**Step**	**Action**
	1.	OS triggers geofence event
	2.	App logs enter event
	3.	Sends anonymized event to backend when possible
Extensions	**Step**	**Branching Action**
	1a	OS denies background location
Sub-variation	**Step**	**Branching Action**
	1	Data fails to send to backend
Related Information		
Priority	High	
Performance	<1 minute	
Frequency	Event-driven	
Channels to actors	Database	
OPEN ISSUES	N/A	
Due Date	December 2025	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

8. Geofencing Leave Detection

Use Case #8	Geofencing Leave Detection	
Goal in Context	Detect when phone leaves lot boundary	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	User previously entered a lot	
Success End Condition	Updated status sends to backend	
Failed End Condition	False positives	
Primary, Secondary Actors	**Primary:** Users (Students and faculty)	
Trigger	Device exits geofence	
Description	**Step**	**Action**
	1.	App detects exit/dwell+walk pattern
	2.	Log leave event
	3.	Send to backend
Extensions	**Step**	**Branching Action**
	1a	False positives Backend filters with dwell patterns
Sub-variation	**Step**	**Branching Action**
	1	Exits are not sent to the backend
Related Information		
Priority	High	
Performance	<1 minute	
Frequency	Event-driven	
Channels to actors	N/A	

OPEN ISSUES	Database
Due Date	December 2025
…any other management info	N/A
Superordinates	N/A
Subordinates	N/A

9. Background Activity Monitoring

Use Case #9	Background Activity Monitoring	
Goal in Context	Maintain accurate sensing while app is in background	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	App has background permissions	
Success End Condition	Batch events to save battery	
Failed End Condition	OS limits background usage	
Primary, Secondary Actors	**Primary:** Users (Students and faculty)	
Trigger	User drives to campus and parks	
Description	**Step**	**Action**
	1.	System uses activity transitions
	2.	Detects driving
	3.	Walking to infer parked state
	4.	Batch events to save battery
Extensions	**Step**	**Branching Action**

	1a	OS limits background Degrade gracefully (log events when opened)
Sub-variation	**Step**	**Branching Action**
	1	Background permissions not allowed
Related Information		
Priority	High	
Performance	<1 minute	
Frequency	Continuous	
Channels to actors	OS Background Service	
OPEN ISSUES	N/A	
Due Date	December 2025	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

10. Anonymous Event Logging

Use Case #10	Anonymous Event Logging
Goal in Context	Collect minimal data for privacy
Scope & Level	**Scope:** SharkPark **Level:** Primary
Preconditions	App sends events
Success End Condition	No user id is stored
Failed End Condition	Vulnerable data is exposed

Primary, Secondary Actors	**Primary:** Backend	
Trigger	Event received by backend	
Description	**Step**	**Action**
	1.	Backend stores event: lot ID
	2.	Enter/leave
	3.	Timestamp
	4.	Reliability tag
	5.	No user id stored
Extensions	**Step**	**Branching Action**
	1a	Malicious data Rate limit and validation
Sub-variation	**Step**	**Branching Action**
	1	Data gets exposed
Related Information		
Priority	High	
Performance	<1 minute	
Frequency	Continuous	
Channels to actors	Database	
OPEN ISSUES	Possible security issues	
Due Date	December 2025	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

11. Real Time Aggregation

Use Case #11	Real Time Aggregation	
Goal in Context	Aggregate events to estimate occupancy	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	Events incoming	
Success End Condition	Estimate occupancy and expose via API	
Failed End Condition	Flagged low confidence	
Primary, Secondary Actors	**Primary:** Backend	
Trigger	Backend scheduled aggregation	
Description	**Step**	**Action**
	1.	Aggregate events per lot
	2.	Compute occupancy estimate
	3.	Expose via API
Extensions	**Step**	**Branching Action**
	1a	Low sample rate Flag low confidence
Sub-variation	**Step**	**Branching Action**
	1	Incorrect estimate
Related Information		
Priority	High	
Performance	<30 seconds	
Frequency	Continuous	
Channels to actors	Database	

OPEN ISSUES	N/A
Due Date	December 2025
…any other management info	N/A
Superordinates	N/A
Subordinates	N/A

12. Short Term Forecasts

Use Case #12	Short Term Forecasts	
Goal in Context	Forecast occupancy 30-60 minutes ahead	
Scope & Level	**Scope:** SharkPark **Level:** Secondary	
Preconditions	Sufficient recent events	
Success End Condition	Store and expose results	
Failed End Condition	Model fails	
Primary, Secondary Actors	**Primary:** ML	
Trigger	System schedules it	
Description	**Step**	**Action**
	1.	ML model ingests recent events
	2.	Outputs forecasts and confidence intervals
	3.	Store and expose results
Extensions	**Step**	**Branching Action**
	1a	Model fails Falls back to historical average

Sub-variation	Step	Branching Action
	1	Insufficient data to create a solid forecast
Related Information		
Priority	High	
Performance	<30 seconds	
Frequency	Regular	
Channels to actors	Database, Scheduled Job	
OPEN ISSUES	Exact ML type and retaining frequency	
Due Date	December 2025	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

13. Long Term Forecasts

Use Case #13	Long Term Forecasts
Goal in Context	Analyze daily/weekly patterns
Scope & Level	**Scope:** SharkPark **Level:** Primary
Preconditions	Historic data available
Success End Condition	Produce long term forecast and trend visualizations
Failed End Condition	Low data reliability
Primary, Secondary Actors	**Primary:** ML
Trigger	Analytic job runs

Description	Step	Action
	1.	Train model on historical patterns
	2.	Produce long term forecasts and trend visualizations
Extensions	**Step**	**Branching Action**
	1a	Insufficient history Show “low data reliability”
Sub-variation	**Step**	**Branching Action**
	1	Incorrect estimate
Related Information		
Priority	Medium	
Performance	<1 hour	
Frequency	Daily/Weekly	
Channels to actors	Database, Scheduled Job	
OPEN ISSUES	Requires historical data window and trend visualization	
Due Date	Spring 2026	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

14. Weather Aware Forecasting

Use Case #14	Weather Aware Forecasting	
Goal in Context	Incorporate weather into forecasts	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	Weather API accessible	
Success End Condition	Accurately updated forecasts	
Failed End Condition	API failure	
Primary, Secondary Actors	**Primary:** Backend	
Trigger	Weather change detected	
Description	**Step**	**Action**
	1.	Fetch weather
	2.	Adjust demand model weights
	3.	Update forecasts
Extensions	**Step**	**Branching Action**
	1a	Api failure Fallback to no-weather model
Sub-variation	**Step**	**Branching Action**
	1	Regular forecast only
Related Information		
Priority	Medium	
Performance	<30 seconds	
Frequency	As needed	
Channels to actors	Third part Weather API, Database	

OPEN ISSUES	Which weather provider to integrate
Due Date	Spring 2026
…any other management info	N/A
Superordinates	N/A
Subordinates	N/A

15. Confidence Intervals Display

Use Case #15	Confidence Intervals Display	
Goal in Context	Show forecasts probability bands	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	Forecasts available	
Success End Condition	System displays forecasts with confidence intervals and reliability meter	
Failed End Condition	Insufficient data	
Primary, Secondary Actors	**Primary:** Student **Secondary:** Backend	
Trigger	User views forecast	
Description	**Step**	**Action**
	1.	System displays forecasts with confidence intervals and reliability meter
Extensions	**Step**	**Branching Action**
	1a	No confidence Show “insufficient data”
Sub-variation	**Step**	**Branching Action**
	1	False estimates

Related Information	
Priority	Medium
Performance	<30 seconds
Frequency	On view
Channels to actors	Mobile App UI
OPEN ISSUES	Confidence band visualization type
Due Date	Spring 2025
…any other management info	N/A
Superordinates	N/A
Subordinates	N/A

16. Battery Optimization

Use Case #16	Battery Optimization	
Goal in Context	Minimize battery usage while monitoring	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	App installed	
Success End Condition	Report metrics to monitor battery impact	
Failed End Condition	User Reports high battery use	
Primary, Secondary Actors	**Primary:** Students **Secondary:** Full stack developers	
Trigger	Background monitoring active	
Description	**Step**	**Action**
	1.	Batch events

	2.	Use low-power sensors, limit GPS sampling
	3.	Report metrics to monitor battery impact
Extensions	**Step**	**Branching Action**
	1a	User reports high battery use Provide settings
Sub-variation	**Step**	**Branching Action**
	1	N/A
Related Information		
Priority	High	
Performance	Semi daily	
Frequency	Continuous	
Channels to actors	OS Background Service	
OPEN ISSUES	Acceptable battery drain threshold TBD	
Due Date	Spring 2025	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

17. Best Lot Recommendation

Use Case #17	Best Lot Recommendation
Goal in Context	Recommend the most ideal available lot
Scope & Level	**Scope:** SharkPark **Level:** Primary
Preconditions	Short-term forecast and aggregation have been formed
Success End Condition	Return the most ideal lot to park within

Failed End Condition	Insufficient data or no lots available	
Primary, Secondary Actors	**Primary:** Students	
Trigger	Block reported or lot/spot requested	
Description	**Step**	
	1.	Query the aggregation, forecast, and reliability meter for current lot data
	2.	Choose the most ideal lot given current data
	3.	Return the selected lot
Extensions	**Step**	
	2a	Iterate through options if the first is not available
Sub-variation	**Step**	
	1	N/A
Related Information		
Priority	High	
Performance	<30 seconds	
Frequency	On demand	
Channels to actors	Mobile App UI, Database	
OPEN ISSUES	Recommendation algorithm weighting TBD	
Due Date	Spring 2026	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

18. Navigate to Selected Lot

Use Case #18	Navigate to Selected Lot	
Goal in Context	Provide the user a route to the selected lot	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	User has selected a lot; location services enabled	
Success End Condition	User is redirected to navigation service with directions to the lot	
Failed End Condition	Navigation service fails to open or route not generated	
Primary, Secondary Actors	**Primary:** Students and faculty **Secondary:** Third party Maps API	
Trigger	User taps “Navigate” option for a lot	
Description	**Step**	
	1.	User selects a parking lot from the app
	2.	System queries backend for the lot’s location and availability
	3.	System generates navigation request with destination coordinates
	4.	Mobile app opens external navigation service (Google/Apple Maps) with route to lot
Extensions	**Step**	
	1a	Location permission disabled App prompts user to enable location
Sub-variation	**Step**	
	1	Maps API unavailable Display error message
Related Information		
Priority	Medium	
Performance	<30 seconds	

Frequency	On demand
Channels to actors	Mobile App UI
OPEN ISSUES	Choice of navigation service
Due Date	Spring 2026
…any other management info	N/A
Superordinates	N/A
Subordinates	N/A

19. Respond to Parking Confirmation Ping

Use Case #19	Respond to Parking Confirmation Ping	
Goal in Context	The user responds to a prompt as to if they are parked	
Scope & Level	**Scope:** SharkPark **Level:** Subfunction	
Preconditions	The user is on campus; location services enabled	
Success End Condition	The user has responded either yes or no to the prompt	
Failed End Condition	The prompt is not sent or is not answered by the user	
Primary, Secondary Actors	**Primary:** Students **Secondary:** Backend	
Trigger	The user has been detected to have left the lot/their vehicle	
Description	**Step**	
	1.	System sends a confirmation prompt to user’s device
	2.	User responds “Yes” or “No”
	3.	System records response and updates occupancy database

Extensions	**Step**	
	1a	User ignores or dismisses prompt System assumes low confidence, uses fallback
Sub-variation	**Step**	
	1	User provides inconsistent input System flags response as unreliable
Related Information		
Priority	Medium	
Performance	<1 minute	
Frequency	Occasionally	
Channels to actors	Mobile App UI, Backend API	
OPEN ISSUES	How often to prompt users without being annoying	
Due Date	Spring 2026	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

20. Spots Seen Report

Use Case #20	Spots Seen Report
Goal in Context	Return a report of spots that users have deemed available
Scope & Level	**Scope:** SharkPark **Level:** Primary
Preconditions	Users have reported available spots around them
Success End Condition	A time-sensitive list of reported open spots is returned
Failed End Condition	No report is returned or no spots have been reported open

Primary, Secondary Actors	**Primary:** Backend	
Trigger	The user queries what spots are known to be open	
Description	**Step**	
	1.	User taps “Spots Seen” button
	2.	System prompts user to confirm lot
	3.	System logs report into database and updates occupancy confidence
Extensions	**Step**	
	1a	Location disables System prompts user to enable location or select lot manually
Sub-variation	**Step**	
	1	Invalid entry App requests valid input
Related Information		
Priority	Medium	
Performance	<30 seconds	
Frequency	On demand	
Channels to actors	Mobile App UI, Backend API, Database	
OPEN ISSUES	Validation and weighting of user reports to avoid spam or false data	
Due Date	Spring 2026	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

21. Manual Block Report

Use Case #21	Manual Block Report	
Goal in Context	Allow users to report obstructions	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	User is logged in	
Success End Condition	Report successfully logged in the database and flagged for review or display	
Failed End Condition	Report fails to send	
Primary, Secondary Actors	**Primary:** Students and faculty **Secondary:** Backend	
Trigger	User observes a blocked area and chooses “report block” in the app	
Description	**Step**	
	1.	App prompts user to select log
	2.	User provides short description
	3.	System marks lot as “partially restricted”
Extensions	**Step**	
	1a	Location not found App prompts user to manually set lot
Sub-variation	**Step**	
	1	Status update fails System retries
Related Information		
Priority	Medium	
Performance	<1 minute	

Frequency	Occasional
Channels to actors	Mobile app UI, Database
OPEN ISSUES	Determine retention time
Due Date	Spring 2026
…any other management info	N/A
Superordinates	N/A
Subordinates	N/A

22. Flag Incorrect Data

Use Case #22	Flag Incorrect Data	
Goal in Context	Enable user to flag obviously incorrect data to improve accuracy	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	User is authenticated	
Success End Condition	Flag stored in backend for validation	
Failed End Condition	Flag submission fails or is ignored	
Primary, Secondary Actors	**Primary:** Students and faculty **Secondary:** Backend	
Trigger	User notices inaccurate data and taps “flag incorrect”	
Description	**Step**	**Action**
	1.	App sends flag event containing lot ID and current occupancy snapshot
	2.	Backend validates (rate limits per user/lot/time)
	3.	System reduces confidence weight for the estimate window and queues review

	4.	If confirmed by multiple users/signals, aggregation re-computes
Extensions	**Step**	**Branching Action**
	1a	<condition causing branching> <action or name of sub use case>
Sub-variation	**Step**	**Branching Action**
	1	N/A
Related Information	N/A	
Priority	Medium	
Performance	<3 seconds	
Frequency	Occasional	
Channels to actors	Mobile UI, Backend API	
OPEN ISSUES	Thresholds for auto-downgrade vs ignore	
Due Date	Spring 2026	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

23. Carpool Detection

Use Case #23	Carpool Detection
Goal in Context	Detect if two or more users enter a lot within the same car
Scope & Level	**Scope:** SharkPark **Level:** Primary
Preconditions	Background sensing active, events arriving.
Success End Condition	Events from a co-moving cluster collapse to a single car arrival/departure

Failed End Condition	Two or more users have been falsely (not) detected as a carpool	
Primary, Secondary Actors	**Primary:** User **Secondary:** Backend	
Trigger	Two or more users enter/exit a lot together	
Description	**Step**	
	1.	Backend computes co-movement features
	2.	If above threshold, mark events as a carpool cluster
	3.	Emit one normalized care event into aggregation
Extensions	**Step**	
	1a	Ambiguous similarity → keep separate but lower confidence
Sub-variation	**Step**	
	1	
Related Information		
Priority	Medium	
Performance	<2 minutes	
Frequency	Each time two or more users enter/exit a lot together	
Channels to actors	Backend API	
OPEN ISSUES	Methodology for deciding if multiple users are carpooling	
Due Date	Spring 2026	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

24. Penetration Rate Estimation

Use Case #24	Penetration Rate Estimation	
Goal in Context	Estimate app user coverage per lot/time and scale counts	
Scope & Level	**Scope:** SharkPark **Level:** Primary	
Preconditions	Application is used by a user within a parking lot	
Success End Condition	Successive operations will account for estimated behavior	
Failed End Condition	Successive operations will only account for app users	
Primary, Secondary Actors	**Primary:** Backend	
Trigger	User with the application parks on campus	
Description	**Step**	
	1.	Account for the actions of the current user (parking, leaving a lot, etc.)
	2.	Calculate behavior averages and scale them to the sizes of each lot in which they occur
	3.	Utilize result for future actions e.g. forecasting
Extensions	**Step**	
	1a	N/A
Sub-variation	**Step**	
	1	N/A
Related Information		
Priority	High	
Performance	<30 seconds	
Frequency	Each time a user with the app enters/leaves a parking lot	

Channels to actors	Backend API
OPEN ISSUES	Accounting for scaling ratios, outlying parking behaviors
Due Date	Fall 2025
…any other management info	N/A
Superordinates	N/A
Subordinates	N/A

25. Reliability Meter Computation

Use Case #25	Reliability Meter Computation	
Goal in Context	computes and assigns confidence levels (High/Medium/Low)	
Scope & Level	**Scope:** SharkPark **Level:** Subfunction	
Preconditions	Occupancy data exist in the database and system has access to timestamps.	
Success End Condition	Confidence levels (High/Medium/Low) is computed and displayed to users in real-time	
Failed End Condition	System is unable to compute confidence levels	
Primary, Secondary Actors	**Primary:** System **Secondary**: Database	
Trigger	User requests parking information and periodic computation	
Description	**Step**	
	1.	System retrieves newest parking data record
	2.	System queries historical metrics and evaluates data reliability
	3.	System computers confidence score and categorize into High/Medium/Low
	4.	System assign visual indicators for the calculated

		confidence score
Extensions	**Step**	
	1a	N/A
Sub-variation	**Step**	
	1	N/A
Related Information		
Priority	High	
Performance	<5 seconds	
Frequency	<15 minutes>	
Channels to actors	Database	
OPEN ISSUES	Adjusted confidence for special events and time of day patterns	
Due Date	Spring 2026	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

26. Shuttle / Transit Integration

Use Case #26	Shuttle / Transit Integration
Goal in Context	Show shuttles/ETAs as an alternative when lots are full
Scope & Level	**Scope**: System **Level**: User-goal
Preconditions	Destination is known and real-time shuttle/API is active and accessible
Success End Condition	Student can view shuttle/transit alternatives

Failed End Condition	Shuttle data unable to load and system reverts to parking-only results	
Primary, Secondary Actors	**Primary:** students **Secondary:** shuttle/transit API	
Trigger	Target lots are at >90% capacity	
Description	**Step**	
	1.	Student enters destination in parking app search
	2.	System detects full lot
	3.	System suggests shuttle alternative
Extensions	**Step**	
	1a	N/A
Sub-variation	**Step**	
	1	
Related Information		
Priority	High	
Performance	<3 seconds	
Frequency	Lots exceed >90%	
Channels to actors	Database, Shuttle/transit API	
OPEN ISSUES	Limited shuttle capacity	
Due Date	Spring 2026	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

27. Dynamic Geofence Management

Use Case #27	Dynamic Geofence Management	
Goal in Context	Rotate and manage active geofences dynamically to maintain coverage accuracy while staying under platform geofence limits	
Scope & Level	**Scope:** SharkPark **Level:** System/Auto	
Preconditions	App has location permissions and geofencing is active	
Success End Condition	System maintains optimal geofence coverage without exceeding OS limits	
Failed End Condition	Some lots are temporarily unmonitored due to limit or permission issues	
Primary, Secondary Actors	**Primary:** System **Secondary:** OS Location Services	
Trigger	System detects active geofence limit approaching or periodic rotation interval reach	
Description	**Step**	
	1.	System evaluates current geofence usage and nearby lot relevance
	2.	Identify less active or out of range lots for rotation and remove low priority geofences
	3.	Confirm successful updates with OS geofencing service
Extensions	**Step**	
	1a	OS denies geofence registration Retry with reduced number or defer until app foregrounded
Sub-variation	**Step**	
	1	Network unavailable Cache pending geofence updates for next cycle

Related Information	
Priority	High
Performance	<1 minute
Frequency	Continuous
Channels to actors	OS Background Service, Backend API
OPEN ISSUES	Determine optimal geofence
Due Date	Spring 2026
…any other management info	N/A
Superordinates	N/A
Subordinates	N/A

28. Event-Aware Forecasting

Use Case #28	Event-Aware Forecasting	
Goal in Context	Adjust predictions using campus event schedules	
Scope & Level	**Scope:** SharkPark **Level:** Subfunction	
Preconditions	Forecasting data is available and busyness of event is known	
Success End Condition	Forecasting takes into account any current events	
Failed End Condition	Forecasting does not take any events into account	
Primary, Secondary Actors	**Primary:** Backend	
Trigger	An event is inputted into the application	
Description	**Step**	
	1.	Include perceived or prior parking-related data for an event that will occur in forecasting operation

	2.	Execute forecasting operation as per normal
	3.	Return forecast alongside a notification of the event
Extensions	**Step**	
	1a	Event data unavailable System continues using base forecast model
Sub-variation	**Step**	
	1	Conflicting event data Apply conservative adjustments and flag low confidence
Related Information		
Priority	Medium	
Performance	<30 seconds	
Frequency	Per forecasting-related operation (e.g. lot querying)	
Channels to actors	Backend API	
OPEN ISSUES	Unknown busyness of events, lack of event parking data	
Due Date	Spring 2026	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

29. Cold-Start Bootstrap Mode

Use Case #29	Cold-Start Bootstrap Mode
Goal in Context	Account for a small user base during early development for data-dependent operations
Scope & Level	**Scope:** SharkPark **Level:** Primary

Preconditions	User has installed the application	
Success End Condition	Operations such as forecasts function for a small user base	
Failed End Condition	Data-dependent functions behave incorrectly	
Primary, Secondary Actors	**Primary:** ML **Secondary:** Backend	
Trigger	User performs an operation such as checking lot capacities	
Description	**Step**	
	1.	Initialize backend with predefined values
	2.	Operate ML model off of predefined values
Extensions	**Step**	
	1a	If first usage, process a stream of predefined values
Sub-variation	**Step**	
	1	
Related Information		
Priority	High	
Performance	<30 seconds	
Frequency	Upon installation	
Channels to actors	Backend API	
OPEN ISSUES	What initial values to choose	
Due Date	Fall 2025	
…any other management info	N/A	
Superordinates	N/A	
Subordinates	N/A	

Functional and Non-Functional Requirements

Functional Requirements

- **Login:** Students login with CSULB email.
- **Parking Map:** Interactive map of all parking lots which are color coded to show occupancy.
- **Live Occupancy:** Shows % full in real time and updates often.
- **Best & Backup Lot:** Suggests best lot to park in and offers backup option.
- **Notifications:** Alerts when lots are almost full or when spots open up.
- **Anonymous Logging:** Only saves lot entered/lot exited with time stamp. No exact location, paths taken or personal data is saved.
- **Forecasting:** Predicts occupancy of lots ahead of time. This makes predictions in the short-term(30-60mins) and long-term(daily/weekly parking patterns).
- **Weather & Events:** Adjusts forecasts given changes in weather or expected campus events.
- **User Feedback:** Students can correct wrong lot info or report accidents, construction, etc.
- **Crowdsourcing:** "Spots seen" button so students can report open spaces.
- **Transit Integration**: Shows shuttle stops and times.
- **Navigation:** One-tap directions to chosen lot via Google/Apple maps.
- **Personalization:** Students can favorite certain lots, set alerts and use light/dark mode.
- **Special Modes:** Detect carpooling, handle early/low usage, show reliability levels.

Non-functional Requirements

Quality Requirements:

- **Accuracy:** Parking lot information should be accurate most of the time. Forecasts should have confidence ranges.
- **Privacy:** No storing user location history.
- **Battery Friendly:** Should not drain phone battery heavily while running in background.
- **Scalability:** Should handle thousands of students at once.
- **Usability:** Easy to use, simple interface with accessible design.
- **Cross-Platform:** Works consistently on iOS and Android.
- **Speed:** Map should load quickly, within 3 seconds. Data should update within 1 minute.

Constraints:

- **Operating System:** iOS (initial release), Android (planned)
- **Languages/Tools:**
    - **Frontend:** React Native using TypeScript, integrating native modules for geofencing (iOS CoreLocation, Android FusedLocation API)

- **Backend:** NestJS (TypeScript, Node.js) for modular, scalable API design
- **Database & Storage:** AWS DynamoDB (real-time events, occupancy data) and AWS S3 (archival logs, model data, static assets)
- **Machine Learning:** AWS SageMaker for training and serving short-term and long-term forecasting models
- **Hosting & Infrastructure:** AWS Lambda (serverless functions), API Gateway, and CloudWatch for monitoring
- **Maps & Geolocation:** Apple Maps SDK (iOS) and Google Maps SDK (Android) for displaying parking lots and navigation routes
- **Notifications:** APNs for iOS and Firebase Cloud Messaging (FCM) for Android push alerts
- **Authentication:** Email-based login using CSULB student credentials; backend-issued JWT tokens for secure session handling
- **Dev & CI/CD:** GitHub (monorepo), GitHub Actions (CI/CD), Docker & Docker Compose (local dev, CI builds)

Development Process

- **Deadlines:**
  - Working demo connecting frontend and backend with simulated occupancy updates by end of Fall 2025.
  - Fully integrated real-time system with predictive and community features by end of Spring 2026
- **Deliverables:**
  - **Fall 2025 — Prototype**
    - Interactive campus map with all parking lots(polygons/IDs/capacities
    - Lot sheet showing occupancy buckets (low, medium, high), no % needed
    - Best lot recommendations based on occupancy
    - Geofence entry/leave detection (iOS CoreLocation/Android Fused Location)
    - Background activity heuristic (drive → walk) for “parked” vs “passing through”
    - Anonymous event logging: (lotID, enter/leave, timestamp)
    - Notifications: near-full/clearing alerts for locally saved favorites.
    - Privacy: raw GPS not stored; raw events auto-purged(e.g., 7 days)
    - Real time aggregation updates lot status within
  - **Spring 2026 — Full working version**
    - Live occupancy map with confidence levels
    - Forecasting with events and weather
    - Battery-optimized sensing
    - Crowdsourcing & reporting tools

- Carpool mode
- Shuttle integration

Conceptual Data Table

Entity	Relationship	Related Entities
Student	Interfaces with	Notification, Map, Reporter, Tracker
Notification	Sent to	Student
Map	Displays	Lot, Route
Map	Transmits data to	User, Notification, Route
Lot	Contains	Spot
Spot	Received from	Reporter, Forecaster
Route	Requested by	User
Reporter	Receives data from	User
Reporter	Transmits data to	Notification, Map, Lot, Spot
Tracker	Follows	User
Tracker	Transmits data to	Notification, Map, Forecaster
Forecaster	Receives data from	Notification, Lot, Spot, Tracker
Forecaster	Transmits data to	Map, Lot, Spot