Team Information

Instructions: Fill in basic information about your team. This helps us track submissions.

Team Name: QuickBites

Team Members (Name, UCINetID):

• Linyu Chen, linyuic

• Diego Nunez, diegon1

• Alex Gonzalez, alexeg1

PROPOSAL CONTENT

1. Project Overview **3**
1.1 Project Title 3
1.2 Problem Statement 3
1.3 Target Users 3
2. System Concept **3**
2.1 High-Level System Description 3
2.2 User Need → System Response 4
3. Data and Information Sources **4**
3.1 Data Sources 4
3.2 Data Collection Strategy 4
4. Personal Model and Context **4**
4.1 Personal Model 4
4.2 Contextual Factors 5
5. Search / Recommendation Logic **5**
5.1 Search or Recommendation Task 5
5.2 Matching and Ranking (Conceptual) 5
6. System Architecture (Conceptual) **5**
6.1 Major Components 5
6.2 User Interface and Presentation 6
7. Project Scope and Feasibility **6**
7.1 What Will Be Implemented 6
7.2 What Is Out of Scope 6
8. Team Plan **6**
8.1 Team Roles and Responsibilities 6
8.2 Anticipated Challenges 7
9. Summary **7**

1. Project Overview

Important: This proposal is exploratory and will evolve throughout the quarter.
You are not expected to have all details finalized.

1.1 Project Title

Instructions: Provide a short, descriptive title that clearly communicates what the system does. (~8–12 words)

Example: Smart EV Charging Advisor for Residential Solar Homes

Smart Lunch Restaurant Search Using Location and Time

1.2 Problem Statement

Instructions: Describe the real-world problem your project addresses and why it matters. Focus on the user's pain point, not the solution. (~120–150 words)

Example: Electric vehicle owners often struggle to decide when to charge their cars in order to minimize cost, reduce grid impact, and preserve battery health. Existing charging solutions are reactive and do not account for personal habits, local energy conditions, or upcoming needs.

Choosing a place to eat lunch is a simple yet frustrating task for many individuals with hectic schedules, particularly college students and working professionals. Finding an affordable, convenient, and satisfying lunch option often requires significant prior knowledge of the surrounding area or time-consuming research. Many people have as little as 30 minutes during a busy workday for lunch and cannot afford to spend much of that time deciding where to eat. As a result, users frequently default to large restaurant chains with prominent advertising, which are not always the healthiest or most suitable options. Current restaurant search solutions often prioritize well-established businesses or venues with large numbers of reviews, while placing less emphasis on convenience, proximity, and local establishments. This bias can prevent users from discovering nearby, affordable, and quick lunch options that better align with their immediate needs and constraints.

1.3 Target Users

Instructions: Describe who the intended users are and the context in which they would use this system. (~60–80 words)

Example: Homeowners with electric vehicles who charge at home and have access to time-of-use electricity pricing

The intended users of our system are busy college students and working professionals with limited time to decide on lunch. These users would search for dining options in their surrounding areas and prioritize results that are affordable, nearby, and accessible. The system is designed for large

environments such as college campuses or dense urban areas with an overwhelming variety of lunch options. The system is intended for mobile use, relying on the user's current location and time.

2. System Concept

2.1 High-Level System Description

Instructions: Explain what kind of system you are building at a conceptual level. Avoid technical details. (~120–150 words)

Example: The system is a personal, context-aware recommendation engine that advises EV owners on optimal charging times. It combines user preferences, historical charging behavior, and external context to proactively suggest actions.

The Smart Lunch Restaurant Search system is a location and time aware lunch recommendation tool that is designed to help users locate and quickly decide where to eat during a limited lunch break. By using the user's current location and time of day, the system automatically identifies nearby restaurants that are realistically reachable and suitable for lunch. It prioritizes options that are convenient, affordable, and fast, helping users avoid wasting time scrolling through overwhelming search results. Additionally, instead of only promoting popular chains or highly reviewed restaurants, the system aims to highlight local and lesser-known businesses that match the user's immediate needs. Users can search on their phone while on campus or at work, and receive a shortlist of practical lunch choices based on distance, estimated travel time, and general accessibility. Overall, the system reduces decision fatigue and supports users in making faster, more satisfying lunch choices.

2.2 User Need → System Response

Instructions: Describe a typical interaction: what triggers the system and what it provides to the user. Important: The system may initiate recommendations without an explicit user query. (~80–100 words)

Example: When the user arrives home and plugs in their vehicle, the system automatically evaluates current grid conditions and recommends whether to charge immediately or delay charging until later in the night.

When a user has a limited lunch break and needs to find a place to eat quickly, they open the system on their phone and allow location access. The system automatically detects the user's current location and current time, then generates a list of nearby lunch options that are reachable within the user's available time. It filters out restaurants that are too far away or likely to take too long, and prioritizes results that are affordable, convenient, and suitable for a quick meal. The user can view the recommended options and select one to get directions or additional details.

3. Data and Information Sources

3.1 Data Sources

Instructions: List and briefly describe all data sources your system relies on. (~100–120 words)

Example: User charging history, user-defined preferences, time-of-use pricing schedules, weather forecasts, and publicly available grid demand data.

The app will rely on several different data sources to generate accurate and useful lunch recommendations. First, it uses the user's real-time location (GPS) to identify restaurants nearby and calculate realistic travel distances. It also uses the current time to focus on restaurants that are open and relevant for lunch hours. The system retrieves restaurant information from online maps and business listing services, including restaurant names, addresses, operating hours, price level, cuisine type, and user ratings. In addition, it may use estimated walking or driving time data to filter out options that are not reachable within a short lunch break. If the user chooses to customize their experience, the system can optionally store user preferences such as budget range, dietary restrictions, and favorite cuisines.

3.2 Data Collection Strategy

Instructions: Explain how each type of data will be collected or obtained. Not everything needs to be real-time. (~80–100 words)

Example: User preferences are entered manually, the application logs charging history, and external data is retrieved via public APIs or static datasets.

The system collects the user's location in real time through mobile GPS permissions when the user opens the application. The current time is gathered automatically from the user's device to determine lunch relevance and filter restaurants by operating hours. Restaurant data such as business names, addresses, price range, ratings, and hours is obtained through a public restaurant or map-based API (such as Google Places or Yelp Fusion). Travel distance and estimated time are retrieved using a navigation or routing API, or calculated based on coordinates. If personalization is enabled, user preferences like budget, cuisine, and dietary needs are entered manually and saved for future searches.

4. Personal Model and Context

4.1 Personal Model

Instructions: Describe how the system models the user and how this affects recommendations. (~100–120 words)

Example: The personal model captures preferred charging windows, sensitivity to cost, and typical driving patterns. These factors influence how aggressively the system delays or advances charging.

The Smart Lunch Restaurant Search system models the user through a combination of real-time context and optional personal preferences. The system automatically considers the user's current location and the time of day to ensure recommendations are nearby, open, and realistic for a lunch break. To better match individual needs, the user can optionally set preferences such as budget range, preferred cuisine types, dietary restrictions (vegetarian, halal, gluten-free, etc.), and transportation method (walking or driving). These factors influence how restaurants are ranked and filtered, allowing the system to prioritize options the user is more likely to enjoy. Over time, the system can also learn from user interactions, such as restaurants they click on or choose frequently, to provide more personalized and relevant lunch recommendations.

4.2 Contextual Factors

Instructions: Identify contextual signals that influence system behavior. (~80–100 words)

Example: Time of day, electricity pricing periods, upcoming travel needs, and current battery level.

The system uses several contextual factors to make its lunch recommendations more practical and relevant. The current time of day is used to focus on lunch-appropriate results and filter restaurants based on whether they are open. The user's real-time location helps the system prioritize restaurants that are nearby and realistically reachable within a short lunch break. Travel distance and estimated travel time (walking or driving) affect which options are shown first. The system may also consider local environment factors such as whether the user is on a campus or in a dense urban area with many choices. If enabled, the user's remaining available lunch time and food preferences further influence which restaurants are recommended.

5. Search / Recommendation Logic

5.1 Search or Recommendation Task

Instructions: Clearly describe what items or actions are being ranked or recommended. (~80–100 words)

Example: Charging start times across a 24-hour window, ranked by cost, convenience, and battery impact.

The system's primary task is to rank and recommend lunch spots that best fit the user's current situation. Using the user's location, time of day, and optional preferences, the system evaluates nearby restaurants and returns a ranked list based on factors such as proximity, estimated travel time, and affordability. Restaurants that are closed, too far from the user, or outside the user's budget are deprioritized. The final output is a concise list of recommendations that balances convenience, relevance, and personal preferences, allowing users to quickly choose a suitable place to eat.

5.2 Matching and Ranking (Conceptual)

Instructions: Explain, at a high level, what makes one option better than another. No algorithms required. (~80–100 words)

Example: Options are ranked higher if they reduce cost, align with user habits, and avoid peak grid demand.

Lunch options are ranked based on how well they fit the user's constraints and preferences. Because time and convenience are major factors during lunch breaks, restaurants that are closer and require shorter travel times are considered more relevant. Options that are open and known for quick service are prioritized over those likely to have long wait times. Affordability plays an important role in ranking, with lower-cost options prioritized by default unless the user specifies a preferred cuisine. To encourage variety and discovery, local and lesser-known restaurants may be promoted over large chains, reducing the dominance of fast-food options and offering users a broader selection of lunch choices.

6. System Architecture (Conceptual)

6.1 Major Components

Instructions: List the main components of your system in plain language. (~5–7 bullets)

Example: Data collection module, personal model, context processor, recommendation engine, user interface.

- **User Interface (Mobile App): Allows users to search for lunch options and view ranked recommendations**

- **Location and Time Context Module: Collects user's current location and time to determine open and nearby locations**
- **Restaurant Data Module: Stores and retrieves restaurant information including location, open hours, price range, and cuisine type**
- **User Preference Module: Manages optional user preferences based on budget, cuisine, and maximum distance**
- **Recommendation Engine: Searches for and ranked restaurants based on proximity, time constraints, and user preferences**

6.2 User Interface and Presentation

Instructions: Describe how recommendations will be presented to the user. (~60–80 words)

Example: A mobile dashboard displaying recommended charging times with brief explanations.

The system presents lunch recommendations through a mobile interface designed for quick and efficient decision-making. Users are shown a ranked list of nearby restaurants, each displaying key information such as distance, price range, cuisine type, and open status. This information is conveyed through clear, compact visual indicators that allow for quick comparison between options. Users can tap on a restaurant to view additional details, including the address and basic directions.

7. Project Scope and Feasibility

7.1 What Will Be Implemented

Instructions: Clearly state what you realistically plan to build this quarter. (~80–100 words)

Example: A functional prototype that generates charging recommendations using simulated data.

Our team plans to build a system that will be able to allow users to see a ranked list of restaurants based on location, time, convenience, and price, with further constraints if the user chooses to customize their search. When the user opens the application, they will be presented with quick questions on what they are looking to eat that day, with a ranked list based on their answers and a search option for added customization.

7.2 What Is Out of Scope

Instructions: List features that are intentionally excluded or left as future work. (~50–70 words)

Example: Direct integration with vehicle hardware or real-time utility billing systems.

Some features that our system will intentionally exclude include real-time wait times or queue lengths since it would require partnerships with local restaurants, in-app food ordering as it would require a significant increase in technical and logistical complexity, and advanced machine learning-based personalization, such as long term behavioral modeling, due to complexity.

8. Team Plan

8.1 Team Roles and Responsibilities

Instructions: Describe how work is divided among team members. (~60–80 words)

Example: One member focuses on data and modeling, another on frontend UI, and another on integration.

This project will divide the work into three roles: system design and recommendation logic, frontend design and UI development, and data sourcing and backend logic. The system design role will focus on implementing the ranking system based on location, time, and user preferences. The frontend role will focus on the mobile interaction and the layout of the recommendations. Finally, the data sourcing role will handle interfacing with restaurant or map-based APIs and managing metadata.

8.2 Anticipated Challenges

Instructions: Identify expected challenges or risks. (~60–80 words)

Example: Limited access to real-world charging data and balancing competing optimization goals.

Some challenges for this project may include balancing relevance and simplicity when presenting information to support quick decision making, APIs having incomplete or outdated information regarding hours, pricing, or cuisine types that lead to incorrect data being presented to the user, accurate time and distance estimations, and ranking tradeoffs when deciding how to balance proximity, affordability, and variety without overly favoring large chains in our search engine.

9. Summary

Instructions: Summarize the project in 2–3 concise sentences. (~50–60 words)

Example: This project proposes a context-aware EV charging advisor that helps homeowners make better charging decisions by combining personal preferences with external data.

This project proposes a context-aware lunch recommendation system designed for busy students and professionals with limited time. By balancing real-time location, time of day, and optional user preferences, the system ranks nearby lunch options based on convenience, affordability, and accessibility. The goal is to reduce decision fatigue and support faster, more satisfying lunch options.