Sprint 1 Planning Document

Sprint Overview:

In this first sprint, the goal is to establish the foundation of the application by implementing user authentication, profile management, and core Anchor functionality. We will build account security features, enable users to create Anchors tied to real-world locations, and configure basic rules such as radius, visibility, expiration, and activation timing. This sprint focuses on setting up the important database tables, creating related backend APIs, and making initial frontend interfaces. We aim to add validation and testing front the first sprint itself to ensure reliable behavior going forward. By the end of the sprint, users should be able to securely create an account, drop Anchors on the map, configure how they appear, and browse nearby Anchors through map and list views, resulting in the first functional version of the platform.

Scrum Master: Pranav Bansal

Meeting Time: Thursdays @ 4:20p, Sundays @ 5pm

Risks and Challenges:

The most significant risk we face in this sprint would be API synchronization. As this is a new project, and having multiple people working on it, everyone has different ways and ideas to structure the backend. Any misalignment between the backend and the frontend could also result in bottlenecks and delays. Additionally, implementing the map and its features could be complex and may consume more time than expected due to none of us having any experience with this before.

User Stories

User Story #1:

As a user, I would like to sign up and log in using email or OAuth so that my Anchors and permissions are securely associated with my identity.

#	Description	Estimated Time	Owner
1	Setting up the Database and Schema for Users	2 Hrs	Pranav
2	Implementing the backend API for Registration and Login with and without OAuth	6 Hrs	Pranav
3	Creating UI for Login and Signup Screens	3 Hrs	Annsh
4	Connecting UI to Backend API Endpoints	3 Hrs	Annsh

Acceptance Criteria:

1. Given that the user enters valid credentials during login, they should be redirected to the main map view.
2. Given that the user selects OAuth, a new account is created or an existing one is logged in.
3. Given that the user enters invalid credentials during login, an error message appears.
4. Given a successful signup, the user data is correctly stored in the database with a hashed password.

User Story #2:

As a user, I would like to reset my password via email so that I can recover my account if I lose access.

#	Description	Estimated Time	Owner
1	Implement password reset API endpoints	3 Hrs	Annsh
2	Build password reset UI flow	3 Hrs	Pranav
3	Implementing the Email system	2 Hrs	Annsh
4	Connecting UI to Backend API endpoints	3 Hrs	Pranav

Acceptance Criteria:

1. Given that the user requests a password reset, they receive a reset link to the provided valid email.
2. Given that the user clicks the reset link, the database updates the hash when they enter a new password.
3. Given that the user tries to login again, the user cannot login with the old password.

User Story #3:

As a user, I would like to log out and revoke sessions so that my account remains secure across devices.

#	Description	Estimated Time	Owner
1	Implementing Token Verification Logic	2 Hrs	Annsh
2	Creating Logout UI	30 mins	Annsh
3	Handling Token Removal	15 Mins	Annsh
4	Automated Unit Tests	30 Mins	Annsh

Acceptance Criteria:

1. Given I am logged in, When I click the logout button, Then I am redirected to the login page and my session is terminated.
2. Given I have logged out, When I attempt to access a protected route or make an API request, Then I receive a 401 Unauthorized response and am redirected to login.
3. Given I complete the logout process, When the logout finishes, Then all authentication tokens are removed from client-side storage and invalidated on the server.
4. Given my token has been revoked, When I try to reuse it on any device, Then access is denied and I must authenticate again.
5. Given I click logout, When the request is processing, Then I see a loading indicator, and when complete, I see a success confirmation message.
6. Given the logout feature is implemented, When automated tests run, Then all unit, integration, and end-to-end tests pass successfully.

User Story #4:

As a user, I would like to manage my identity by updating my username, email, avatar, and bio so that I am recognizable within Circles.

#	Description	Estimated Time	Owner
1	Implement PATCH /user/profile endpoint	2 hrs	Pranav

	to update fields.		
2	Integrate S3 for Avatar image upload.	3 hrs	Shriyan
3	Build "Edit Profile" screen with inputs for Bio, Name, and Image Picker.	3 hrs	Pranav
4	Test that updating the email checks for uniqueness (no duplicate emails).	30 mins	Pranav

Acceptance Criteria:

1. Given a user on the profile page, When they upload a new photo, Then the photo is displayed and persists after app restart.
2. Given a user changes their username, When they save, Then the new username is visible to other users on the map.
3. Given a user tries to change their email to one that already exists, When they submit, Then the system returns a Conflict error.

User Story #5:

As a user, I would like to create an Anchor by dropping a pin on a map so that content is tied to a precise physical location.

#	Description	Estimated Time	Owner
1	Implement the Anchor creation API	4 Hrs	Shriyan
2	Implementing the database schema for Anchors	2 Hrs	Aryan
3	Implement the map service	5 Hrs	Aryan
4	Build anchor creation UI	4 Hrs	Aryan
5	Get the Current Location of the User to tie it to the Anchor’s location	2 Hrs	Aryan
6	Connect backend to frontend	2 Hrs	Shriyan
7	Test if users can see Anchors created by others, edge cases.	2 Hr	Shriyan

Acceptance Criteria:

1. Given that the map is loaded, when the user clicks the "Drop Anchor" button, a pin appears at their current geolocation.
2. Given that an Anchor is created, when the app is restarted, the pin remains at the correct coordinates.

3. Given that the user creates an Anchor, when the request is sent, the backend stores the latitude, longitude, and creator ID.

User Story #6:

As a user, I would like to configure an unlock radius (e.g., 10–100 meters) so that Anchors only unlock when users are physically nearby.

#	Description	Estimated Time	Owner
1	UI for Anchor Radius	1 Hr	Aryan
2	Adding radius parameters to the Anchor database	30 mins	Shriyan
3	Connecting backend with UI	30 mins	Shriyan
4	Automated Unit Tests	30 mins	Shriyan

Acceptance Criteria:

1. Given I am creating or editing an Anchor, When I view the configuration form, Then I see a radius input field with a slider or numeric input allowing me to set a value between 10 and 100 meters.
2. Given I set an unlock radius for my Anchor, When I save the Anchor, Then the radius value is stored in the database and associated with that Anchor.
3. Given I attempt to set a radius outside the valid range, When I try to save, Then I see a validation error message indicating the radius must be between 10 and 100 meters.
4. Given I have configured an Anchor with a specific radius, When I view or edit that Anchor later, Then the previously set radius value is correctly displayed in the UI.
5. Given the radius configuration is implemented, When the frontend submits the radius data to the backend, Then the API successfully receives, validates, and stores the radius parameter.
6. Given the radius feature is complete, When automated tests run, Then all database schema changes, API validations, and UI interactions are verified and pass successfully.

User Story #7:

As a user, I would like to set an Anchor's visibility to public, private, or Circle-based so that sensitive content is protected.

#	Description	Estimated Time	Owner
1	Add visibility Enum column (PUBLIC, PRIVATE, CIRCLE) to Anchor table.	30 mins	Aryan
2	Add visibility dropdown/selector to creation form.	30 mins	Aryan
3	Testing: Create a private anchor and verify	30 mins	Shriyan

	another user cannot fetch it via API.		

Acceptance Criteria:

1. Given User A creates a "Private" anchor, When User B searches the area, Then User B sees nothing.
2. Given User A creates a "Public" anchor, When User B searches the area, Then User B sees the pin on the map.
3. Given an anchor is "Circle-based", When a user not in the circle approaches, Then the anchor remains invisible.

User Story #8:

As a user, I would like to set expiration parameters (time-based expiration or maximum "unlock count") so that the Anchor automatically disappears after a specific duration or once the maximum number of users have successfully unlocked it.

#	Description	Estimated Time	Owner
1	Create UI for Expiration Settings (Time & Count)	2 Hrs	Aryan
2	Implement backend logic to check for expiration	1 Hr	Shriyan
3	Implement backend logic to check for unlock count	30 mins	Shriyan
4	Create Background Job to Clean Up Expired Anchors	1 Hr	Aryan

Acceptance Criteria:

1. Given I am creating or editing an Anchor, When I view the configuration form, Then I see options to set a time-based expiration (date/time or duration) and/or a maximum unlock count.
2. Given I set a time-based expiration for my Anchor, When the expiration time is reached, Then the Anchor is automatically marked as expired and no longer appears to users.
3. Given I set a maximum unlock count for my Anchor, When the number of successful unlocks reaches that limit, Then the Anchor is automatically marked as expired and becomes unavailable.
4. Given I attempt to set invalid expiration parameters (e.g., past date, negative count), When I try to save, Then I see validation error messages indicating the correct format and constraints.
5. Given expired Anchors exist in the system, When the background cleanup job runs, Then all expired Anchors are removed or archived from the active database.
6. Given the expiration feature is complete, When automated tests run, Then all expiration logic (time-based and count-based), validation rules, and background job functionality are verified and pass successfully.

User Story #9:

As a user, I would like to edit or delete the Anchors I created so that outdated or incorrect information can be removed.

#	Description	Estimated Time	Owner
1	Implement edit and delete API endpoints	3 Hrs	Shriyan
2	Build Anchor edit UI with map	4 Hrs	Aryan
3	Connecting backend with UI	1 Hr	Aryan
4	Automated Tests	1 Hr	Shriyan

Acceptance Criteria:

1. Given I am viewing my created Anchor, When I click the edit button, Then I see a pre-populated form with all existing Anchor details and an interactive map showing the current pin location.
2. Given I am editing my Anchor, When I modify the details (title, description, coordinates, radius, expiration) and click save, Then the changes are successfully saved to the database and reflected immediately in the UI.
3. Given I attempt to edit or delete an Anchor I do not own, When the request is sent to the backend, Then I receive a 403 Forbidden response and see an error message indicating I lack permission.
4. Given I am viewing my Anchor, When I click the delete button, Then I see a confirmation dialog asking "Are you sure you want to delete this Anchor?" before proceeding.
5. Given I confirm deletion of my Anchor, When the delete operation completes successfully, Then the Anchor is removed from the database and no longer appears in my Anchor list or on the map.
6. Given the edit and delete features are implemented, When automated tests run, Then all API endpoints, authorization checks, UI interactions, and error handling scenarios are verified and pass successfully.

User Story #10:

As a user, I would like to create tags for Anchors so that people can get a brief idea of what the Anchor is about before clicking on it.

#	Description	Estimated Time	Owner
1	Implement UI for Tags in Anchor Creation and Discovery	2 Hrs	Aryan
2	Implement Search/Filter UI	3 Hrs	Annsh
3	Update the database to support tags	1 Hr	Aryan

4	Implement search/filter logic	2 Hrs	Annsh
5	Automated Tests	1 Hr	Annsh

Acceptance Criteria:

1. Given I am creating or editing an Anchor, When I view the form, Then I see a tag input field where I can add multiple tags to describe the Anchor.
2. Given I am browsing Anchors on the map or in a list, When I view an Anchor preview, Then I see the associated tags displayed clearly before clicking to view full details.
3. Given I am on the Anchor discovery page, When I use the search/filter UI, Then I can filter Anchors by selecting one or multiple tags to narrow down results.
4. Given I search or filter Anchors by specific tags, When the filter is applied, Then only Anchors containing those tags are displayed in the results.
5. Given I save an Anchor with tags, When the data is submitted to the backend, Then the tags are stored in the database and correctly associated with that Anchor.
6. Given the tag feature is complete, When automated tests run, Then all tag creation, storage, display, and search/filter functionality are verified and pass successfully.

User Story #11:

As a user, I would like Anchors to activate only during specific time windows so that they align with real-world events.

#	Description	Estimated Time	Owner
1	Add start/end time fields to Anchor database model (UTC)	2 Hrs	Pranav
2	Add date/time picker in Anchor creation UI (with “always active” option). Validate date/time fields.	4 Hrs	Annsh
3	Prevent unlocking Anchors outside activation window	2 Hrs	Pranav
5	Backend unit tests for time boundary cases	2 Hrs	Annsh

Acceptance Criteria:

1. Given I am creating an Anchor, when I select a start and end time (or choose “Always active”), then the app validates the inputs, saves the correct activation window in UTC, and prevents publishing if the time window is invalid.
2. Given an Anchor has an activation window that is not currently active, when I request nearby Anchors, then the Anchor does not appear in the nearby list or map results until the current time falls within its start and end time window.
3. Given I can see or select an Anchor that is outside its activation window, when I attempt to unlock it, then the unlock is denied and I receive a clear message indicating the Anchor is not currently active.
4. Given backend unit tests are run, when the current time is exactly at the activation start time or activation end time and when the window is invalid, then the system behaves correctly for these boundary cases.

User Story #12:

As a user, I would like to view a “List View” of the nearby Anchors so that I can browse content without solely relying on the map.

#	Description	Estimated Time	Owner
1	Create UI with scrollable list screen with a basic header and reusable row cards: title/tag, distance, lock icon/state, small metadata (public/private/circle) .	3 Hrs	Annsh
3	Sort by distance by default. Lightweight filter (locked/unlocked, private/public)	2 Hrs	Pranav
4	On tap: highlight location on map, show preview based on locked/unlocked	3 Hrs	Annsh
5	Backend support to GET all anchor information	2 Hrs	Shriyan
6	Unit tests for sorting/filtering. Basic UI test that list renders from mock data and tap routes correctly.	3 Hrs	Pranav

Acceptance Criteria:

1. Given I open the Nearby Anchors List View, when the app requests nearby Anchor data, then I see a loading state while fetching, a clear empty state if none exist, and otherwise a scrollable list where each row displays the Anchor's title or tag, distance, lock status, and visibility metadata.
2. Given multiple Anchors are present, when the list is displayed or filters are applied, then Anchors are sorted from closest to farthest by default and update immediately to reflect locked or unlocked and public or private filtering selections.
3. Given I tap an Anchor row in the list, when the selection occurs, then the app highlights that Anchor's location on the map and preserves the selected Anchor state.
4. Given I select an Anchor from the list, when the preview opens, then private Anchors show only limited preview information while public/accessible Anchors display their full accessible content returned from the backend.
5. Given the list and preview features depend on backend data, when the endpoint is called and tests are executed, then all required Anchor fields are returned and sorting, filtering, rendering, and tap navigation behaviors function correctly across normal and edge cases

Product Backlog

Key:
Current Sprint
~~Completed~~
Still Remaining

Functional Requirements

1. **Authentication and Account Security**
    a. As a user, I would like to sign up and log in using email or OAuth so that my Anchors and permissions are securely associated with my identity.
    b. As a user, I would like to reset my password via email so that I can recover my account if I lose access.
    c. As a user, I would like to login using existing safe credentials on my device (e.g., FaceID, Touch ID, etc.). (if time allows)
    d. As a user, I would like to log out and revoke sessions so that my account remains secure across devices.
    e. Authentication tokens shall expire and be revocable.
2. **Identity and Profile**
    a. As a user, I would like to manage my identity by updating my username, email, avatar, and bio so that I am recognizable within Circles.
3. **Anchor Creation and Configuration**
    a. As a user, I would like to create an Anchor by dropping a pin on a map so that content is tied to a precise physical location.
    b. As a user, I would like to attach different content types (text, links, images, files) to an Anchor so that information is flexible and useful.
    c. As a user, I would like to configure an unlock radius (e.g., 10–100 meters) so that Anchors only unlock when users are physically nearby.
    d. As a user, I would like to set an Anchor's visibility to public, private, or Circle-based so that sensitive content is protected.
    e. As a user, I would like to set expiration parameters (time-based expiration or maximum "unlock count") so that the Anchor automatically disappears after a specific duration or once the maximum number of users have successfully unlocked it.
    f. As a user, I would like to edit or delete the Anchors I created so that outdated or incorrect information can be removed.
    g. As a user, I would like to preview how an Anchor will appear to others before publishing it so that I can verify correctness.
    h. As a user, I would like to create tags for Anchors so that people can get a brief idea of what the Anchor is about before clicking on it.
    i. As a user, I would like Anchors to activate only during specific time windows so that they align with real-world events.

j. As a user, I would like to create unsavable Anchors (special tag) so that sensitive, personal information cannot be saved forever.

4. **Anchor Discovery**
    a. As a user, I would like to view nearby Anchors on a map so that I can discover relevant information around me.
    b. As a user, I would like Anchors to automatically unlock when I enter their radius so that I do not need to refresh or search manually.
    c. As a user, I would like Anchors outside my radius to remain hidden or blurred so that location enforcement is preserved.
    d. As a user, I would like to receive notifications when new Anchors appear nearby so that I do not miss relevant updates.
    e. As a user, I would like to filter based on tags, content type, expiration status, etc, so that the map view remains legible in dense areas.
    f. As a user, I would like to view a "List View" of the nearby Anchors so that I can browse content without solely relying on the map. (*if time allows).*
5. **Saving and Personal Library**
    a. As a user, I would like to save Anchors to a personal library so that I can reference that information forever without being in the same location and worrying about an expiration date.
    b. As a user, I should be able to differentiate between Expired and Non-Expired Saved Anchors in my personal library.
6. **Privacy Controls**
    a. As a user, I want to toggle "Ghost Mode" that stops tracking a user's location so that privacy can be maintained if wanted.
    b. As a user, I would like to block users so that I do not see their Anchors. (if time allows)
7. **Circles and Access Control**
    a. As a user, I would like to create Circles (groups) so that I can share Anchors with specific people only.
    b. As a user, I would like to invite or remove members from a Circle so that access control remains accurate.
    c. As a user, I want to see a "Global Search" for public Circles, so that a user can discover public groups.
8. **Social and Engagement**
    a. As a user, I would like to upvote/downvote anchors to provide a signal for the credibility/usefulness of an anchor. (if time allows)
    b. As a user, I want to see how many people have unlocked my Anchor so that social proof is established.
9. **Scavenger and Hunt Games**
    a. As a user, I would like to create "Scavenger hunts" by linking Anchors in a sequential process where unlocking one Anchor reveals the next clue. (if time allows)
    b. As a user, I would like to track my progress through a Scavenger hunt so that I know how many steps remain. (if time allows)

    c. As a user, I would like to view a leaderboard for hunts so that participation is engaging and competitive. (if time allows)

10. **Moderation and Administration**
    a. As a user, I would like to report inappropriate Anchors so that the platform remains safe.
    b. As an admin, I would like to review reported Anchors so that I can take moderation actions.
    c. As an admin, I would like to disable or remove abusive users so that misuse is prevented.
    d. As a system, I would like to maintain an audit log of critical actions so that abuse investigations are possible.
11. **AR Mode (Optional Enhancement)**
    a. As a user, I would like to view Anchors in an AR camera mode so that I can see distance and direction visually. (if time allows)
    b. As a user, I would like AR Anchors to update in real time as I move so that navigation feels accurate. (if time allows)

Non-Functional Requirements

1. **Performance and Latency**
    a. The system shall unlock Anchors within 500 ms after a user enters the configured radius.
    b. Map and nearby Anchor queries shall respond within 1 second under normal load.
2. **Scalability and Load Handling**
    a. The backend shall support ~10,000 concurrent users without degradation.
    b. The system shall be horizontally scalable to handle increasing Anchor density in urban areas.
    c. The spatial backend shall support efficient geospatial queries using indexing.
3. **Availability and Reliability**
    a. The service shall be available 24/7 with at least 99% uptime.
    b. Anchors and permissions data shall be persisted reliably with automated backups.
4. **Security and Privacy**
    a. All Anchor content shall be encrypted at rest and in transit.
    b. Unauthorized users shall never receive decrypted Anchor content.
    c. Authentication tokens shall expire and be revocable.
    d. Location spoofing attempts shall be mitigated using consistency checks.
5. **Abuse Prevention**
    a. The system should rate-limit Anchor creation so that spam is minimized. (if time allows)
6. **Usability and UX**

   a. New users shall be able to create their first Anchor within ~2 minutes of onboarding.
   b. The UI shall be consistent across Android and iOS platforms.
   c. The app shall provide clear visual feedback for locked vs. unlocked Anchors.
7. **Architecture and Extensibility**
   a. The backend shall follow a modular FastAPI architecture.
   b. The system shall support future extensions such as AR HUDs and new Anchor types without major refactors.
8. **Offline and Caching**
   a. The app shall cache metadata for nearby Anchors to allow discovery in areas with poor connectivity. (if time allows)