PROJECT SPECIFICATION: PlantSocial (Working Title)

Version: 1.0 (MVP Phase) **Date:** February 11, 2026

---

1. Executive Summary

PlantSocial is a "vertical social network" designed specifically for plant enthusiasts. Unlike general social media, PlantSocial connects users based on **shared context**: the specific plant species they are growing and their local environmental conditions.

The core value proposition is **"Dynamic Cohorts."** A user growing Tomatoes in a cold climate should see content from other users growing Tomatoes in similar conditions, solving the problem of irrelevant gardening advice found on generic platforms.

2. Market Value & Target Audience

- **The "Why":** Generic advice fails in gardening. "Water twice a week" kills a plant in Arizona but drowns it in Seattle.
- **Target Market:** Urban gardeners, hobbyist growers, and "plant parents" who value data-driven success over generic aesthetic photos.
- **Business Model Potential:** Future monetization through affiliate links for specific tools/fertilizers recommended by the AI or high-ranking users, and premium "Plant Doctor" features.

---

3. Technical Architecture (Strict Constraints)

To ensure employability and portfolio quality, the development team must adhere to the following Enterprise-grade stack.

3.1 Backend

- **Framework:** Spring Boot 3.2+ (Java 21)
- **Database:** PostgreSQL (Hosted on Neon/Render)
- **Security:** Spring Security 6 with Stateless JWT Authentication.
- **API Documentation:** Swagger/OpenAPI (Springdoc) is mandatory.
- **Architecture Pattern:** Controller Service Repository.
- **DTOs:** Use Java Records for all Data Transfer Objects. *Never return Entities directly.*

3.2 Frontend

- **Framework:** Angular 17+
- **Architecture:** Standalone Components (No NgModules).
- **State Management:** Angular Signals.
- **UI Library:** Angular Material or PrimeNG.
- **Styling:** SCSS / Tailwind CSS.

3.3 Infrastructure & APIs

- **Identification API:** Pl@ntNet API (Free Tier).
- **Plant Doctor:** *Phase 2 Feature.* (Currently: Placeholder UI).
- **Image Storage:** Cloudinary or AWS S3 (Free Tier).

---

4. Functional Requirements (MVP Scope)

4.1 User Module (User)

- **Registration/Login:** Secure signup with email/password. JWT token issued on login.
- **Profile:** Display Username, Location (Zone/City), and "Green Thumb Score" (Streak count).
- **Streaks:** Logic to track consecutive days of activity. If a user misses a day (based on UTC midnight), the streak resets.

4.2 The "My Garden" Module (UserPlant)

- **Add Plant:** User uploads a photo Sent to **Pl@ntNet API** Returns Plant Name Saved to Database.
- **Plant Dashboard:** List of user's active plants.

4.3 Social Feed Module (Post)

- **Dynamic Sub-feeds:**
    - Clicking "Tomato" in My Garden opens the global "Tomato Feed."
    - *Constraint:* Posts must be filterable by Plant ID.

- **Posting:** Users can post an update (photo + text) linked to a specific plant in their garden.

4.4 Gamification Module (Vote)

- **Leaderboards:** A monthly view showing the "Top Harvests."
- **Voting Logic:**
    - Every user gets exactly **3 votes per month**.
    - Users cannot vote for themselves.
    - Backend must validate count(votes) <= 3 before accepting a request.

---

5. Developer Standards & Workflow

To simulate a professional environment, developers must follow these rules:

1. **Git Flow:**
    - main branch is protected.
    - Feature branches must be named feat/feature-name (e.g., feat/auth-login).
    - Pull Requests (PRs) require at least one approval.
2. **Code Quality (SOLID):**
    - **S:** Single Responsibility. A PlantService should not handle User Authentication.
    - **D:** Dependency Injection. Always inject repositories via constructor injection.
3. **Security:**
    - **NEVER** commit API keys or DB passwords to GitHub. Use application.properties with environment variables (e.g., ${DB_PASSWORD}).

---

6. Implementation Roadmap

Week	Focus Area	Backend Tasks (Spring Boot)	Frontend Tasks (Angular 17+)	Critical Milestone
Week 1	Foundation & Auth	• Init Spring Boot + PostgreSQL (Neon). • Create User Entity & Repository. • Implement JWT Auth (Login/Signup endpoints). • Deploy DB & API to Render (Day 3).	• Init Angular (Standalone). • Setup UI Lib (Material/PrimeNG). • Build Login/Signup Forms. • Create AuthInterceptor (attach JWT). • Connect FE to deployed BE.	"Hello World" Auth: A user can sign up and log in on the live site.
Week 2	Core Features (The "Garden")	• Create Plant & Post Entities. • Integrate Pl@ntNet API Service.	• Build "My Garden" Dashboard (List View). • "Add Plant" Component (Camera/Upload).	Core Loop: A user can upload a photo, get it identified, and see it in their dashboard.

Week	Focus Area	Backend Tasks (Spring Boot)	Frontend Tasks (Angular 17+)	Critical Milestone
		• Endpoint: POST /api/plants (Upload + ID). • Endpoint: GET /api/feed/{plantName}.	• "Plant Feed" Component (The dynamic subreddit). • Test Image Upload flow.	
Week 3	Social & Polish	• Leaderboard Logic: SELECT top harvests. • Voting System: 3-vote limit logic. • Optimize DB queries (Indexing). • Final Security Audit (CORS/Env Vars).	• Leaderboard UI (Grid of best plants). • Voting Buttons & Interaction logic. • Global Styling/CSS Cleanup. • Final Deployment to Vercel.	Demo Ready: The app is live, looks good, and the main "Vote & Grow" loop works.

Action Items for Developers

1. **Clone the Repository:** https://github.com/PhantomVisible/PlantSocial
2. **Review the Stack:** Ensure you have Java 21 and Node.js installed.

3. **Wait for Task Assignment:** Tasks will be distributed via Jira.