GitHop: Schedule Feasibility Report

Project Timeline and Implementation Plan

October 7, 2025

Executive Summary

This schedule feasibility analysis confirms that the GitHop project can be successfully completed within a 6-week timeline (approximately 1.5 months). The project is broken down into 7 major phases with clear deliverables and dependencies. The timeline accounts for parallel work streams where possible, with the most critical path being the backend development and data integration. Key milestones include completing the MVP by Week 4 and final deployment by Week 6. The schedule is aggressive but achievable with focused effort.

1 Project Overview and Timeline Framework

1.1 Project Scope

GitHop is a web application that provides trending GitHub activity through a social feed interface, featuring:

- Top explored and fast-growing repositories
- Most active developers by field
- Popular topics and trending technologies
- Social feed-style user experience with advanced filtering

1.2 Timeline Constraints

- **Total Duration:** 6 weeks (1.5 months)
- **Team Size:** 2 developers
- **Workload:** Partial-time equivalent (20-25 hours/week)

Table 1: Phase 1: Foundation Tasks

Task	**Duration**	**Dependencies**
Environment setup (Dev, Staging, Production)	1 day	None
Technology stack finalization	0.5 day	None
Project repository initialization	0.5 day	Environment setup
Basic CI/CD pipeline setup	1 day	Repository setup
API research & selection (GitHub API, ranking APIs)	2 days	None

2 Project Breakdown: 7-Phase Approach

2.1 Phase 1: Project Setup & Foundation (Week 1)

Duration: 5 days
Critical Dependencies: None

Deliverables:

- Development environment ready
- Project repository with initial structure
- CI/CD pipeline operational
- API documentation and access keys

2.2 Phase 2: Data Architecture & API Integration (Week 1-2)

Duration: 6 days
Critical Dependencies: Phase 1 completion

Table 2: Phase 2: Data Architecture Tasks

Task	**Duration**	**Dependencies**
Database design & schema creation	2 days	Technology stack
GitHub API integration setup	2 days	API research
Data collection services development	3 days	Database design
Data caching strategy implementation	1 day	API integration
Data aggregation logic for trends	2 days	Data collection

Deliverables:

- Database schema and initial data models
- Functional GitHub API integration
- Basic data collection services
- Data aggregation pipeline

2.3 Phase 3: Core Backend Development (Week 2-3)

Duration: 8 days
Critical Dependencies: Phase 2 completion

Table 3: Phase 3: Backend Development Tasks

Task	Duration	Dependencies
API endpoint development	3 days	Data architecture
Trending algorithm implementation	3 days	Data aggregation
Filter system backend	2 days	API endpoints
Authentication & authorization	2 days	API endpoints
Data update scheduling system	1 day	Trending algorithm

Deliverables:

- Complete REST API with endpoints
- Trending calculation algorithms
- Backend filtering system
- User authentication system

2.4 Phase 4: Frontend Foundation (Week 2-3)

Duration: 7 days
Critical Dependencies: Can start parallel with Phase 3 after initial API design

Table 4: Phase 4: Frontend Foundation Tasks

Task	Duration	Dependencies
UI/UX design finalization	2 days	None
Frontend framework setup	1 day	Technology stack
Basic layout & component structure	2 days	UI/UX design
API integration on frontend	2 days	Backend API endpoints
State management setup	1 day	Framework setup

Deliverables:

- Complete UI/UX design mockups
- Frontend application structure
- Basic components and layout
- API integration layer

2.5 Phase 5: Core Features Implementation (Week 3-4)

Duration: 8 days
Critical Dependencies: Phase 3 & 4 completion

Table 5: Phase 5: Core Features Implementation

Task	**Duration**	**Dependencies**
Social feed component development	3 days	Frontend foundation
Repository listing & detail views	2 days	API integration
Developer profile components	2 days	API integration
Topic/field categorization system	2 days	Backend data
Basic search functionality	1 day	Backend filters

Deliverables:

- Functional social feed interface
- Repository exploration features
- Developer activity views
- Topic categorization

2.6 Phase 6: Advanced Features & Filter System (Week 4-5)

Duration: 7 days
Critical Dependencies: Phase 5 completion

Table 6: Phase 6: Advanced Features Implementation

Task	**Duration**	**Dependencies**
Advanced filtering system (CRITICAL)	3 days	Core features
Dashboard charts & visualization	2 days	Core features
Combined view implementation	2 days	All components
Mobile responsiveness optimization	2 days	Core features
Performance optimization	1 day	All features

Deliverables:

- Complete filtering system (100% requirement from survey)
- Data visualization components
- Combined presentation style (62.5% survey preference)
- Mobile-responsive design

Table 7: Phase 7: Final Testing and Deployment

Task	**Duration**	**Dependencies**
Comprehensive testing (unit, integration)	2 days	All features
User acceptance testing & bug fixes	2 days	Comprehensive testing
Production deployment	1 day	Testing completion
Documentation & user guides	1 day	Deployment
Performance monitoring setup	0.5 day	Deployment
Launch announcement & initial promotion	0.5 day	Deployment

2.7 Phase 7: Testing, Deployment & Launch (Week 5-6)

Duration: 6 days
Critical Dependencies: All previous phases
Deliverables:

- Fully tested application
- Production deployment
- User documentation
- Live GitHop application

3 Project Timeline Visualization

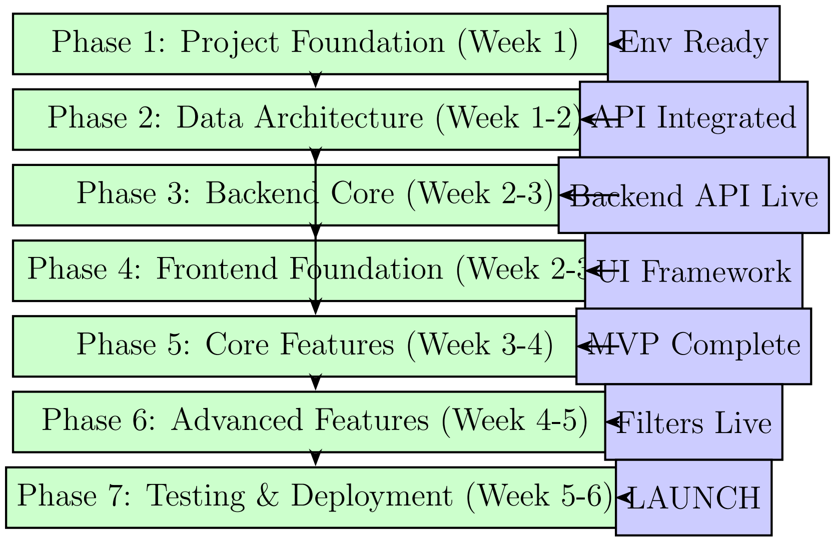
Phase 1: Project Foundation (Week 1)
Env Ready
Phase 2: Data Architecture (Week 1-2)
API Integrated
Phase 3: Backend Core (Week 2-3)
Backend API Live
Phase 4: Frontend Foundation (Week 2-3
UI Framework
Phase 5: Core Features (Week 3-4)
MVP Complete
Phase 6: Advanced Features (Week 4-5)
Filters Live
Phase 7: Testing & Deployment (Week 5-6)
LAUNCH

Figure 1: GitHop Project Timeline with Key Milestones

4 Risk Analysis and Mitigation Strategies

4.1 Potential Risks and Solutions

Table 8: Risk Assessment and Mitigation

Risk	**Impact**	**Mitigation Strategy**
GitHub API rate limiting	High - Could affect data collection	Implement caching, use multiple tokens, fallback to external datasets
Complex trending algorithms	Medium - Could delay backend	Start with simple algorithms, iterate later
Filter system complexity	High - Critical feature from survey	Prioritize basic filters first, advanced later
Frontend performance with large datasets	Medium - Could affect user experience	Implement virtual scrolling, pagination, optimize API calls
Tight timeline	High - Could affect quality	Focus on MVP features, defer nice-to-haves to post-launch

5 Feasibility Conclusion

5.1 Schedule Viability Assessment

Based on the detailed breakdown, the GitHop project is **highly feasible** within the 6-week timeline due to:

- **Clear Phase Dependencies:** Well-defined sequential and parallel work streams
- **Aggressive but Realistic Estimates:** Each phase has buffer time for unexpected delays
- **MVP-Focused Approach:** Core functionality can be delivered by Week 4
- **Risk Mitigation:** Identified risks have clear mitigation strategies

5.2 Critical Success Factors

- **Filter System Priority:** Must complete filtering by Week 5 (100% user requirement)
- **API Integration:** Data collection must be stable by Week 2
- **Combined Presentation:** Social feed + dashboard + lists by Week 5 (62.5% user preference)

5.3 Recommendation

Proceed with the project implementation following the 7-phase approach. The timeline is tight but achievable with focused effort, and delivers all critical features identified in user research within the 1.5-month constraint.