Grasp - Complete Development Workflow

📋 Project Overview

Timeline: 10-12 weeks

Daily Commitment: 2-4 hours

Skill Level: Intermediate JavaScript

🚀 Phase 1: Project Foundation (Week 1)

Day 1: Initial Setup

- [ ] Create GitHub repository named "grasp-news"
- [ ] Install Node.js (v18 or later) if not installed
- [ ] Create Next.js project: `npx create-next-app@latest grasp --use-npm --no-typescript`
- [ ] Navigate to project: `cd grasp`
- [ ] Initialize git and make first commit
- [ ] Create `.env.local` file for environment variables
- [ ] Update `README.md` with project description

Day 2: Dependencies Installation

- [ ] Install core dependencies:

```bash
npm install @supabase/supabase-js @supabase/auth-helpers-nextjs
npm install axios date-fns
npm install zustand
```

- [ ] Install UI dependencies:

```bash
npm install @headlessui/react @heroicons/react
npm install framer-motion
npm install react-hot-toast
```

- [ ] Install map dependencies:

```bash
npm install leaflet react-leaflet
npm install -D @types/leaflet
```

- [ ] Install chart dependencies:

bash

```
npm install chart.js react-chartjs-2
```

Day 3: Project Structure

- [ ] Create folder structure:

```
/app
  /api
  /auth
  /news
  /profile
  /components
/components
  /ui
  /news
  /map
  /charts
/lib
/hooks
/utils
/public
  /images
```

- [ ] Set up TailwindCSS properly (should be auto-configured)
- [ ] Create `jsconfig.json` for path aliases:

json

```
{
  "compilerOptions": {
    "paths": {
      "@/*": ["./*"]
    }
  }
}
```

Day 4: Design System Setup

- [ ] Create color palette in `app/globals.css`:
  - Primary: Blue (`#3B82F6`)
  - Secondary: Green (`#10B981`)
  - Neutral: Gray shades
  - Sentiment colors: Red/Yellow/Green
- [ ] Set up font system (Inter or similar)

- [ ] Create CSS variables for consistent spacing
- [ ] Set up dark mode CSS variables

Day 5: Basic Components

- [ ] Create `components/ui/Button.js`
- [ ] Create `components/ui/Card.js`
- [ ] Create `components/ui/Input.js`
- [ ] Create `components/ui/Badge.js`
- [ ] Create `components/ui/Skeleton.js` (for loading states)
- [ ] Create `components/Layout.js` (header, footer, navigation)

Day 6: Navigation & Routing

- [ ] Create `components/Header.js` with logo and navigation
- [ ] Create `components/Footer.js`
- [ ] Set up main pages structure:
  - `app/page.js` (Homepage)
  - `app/news/page.js` (News list)
  - `app/news/[id]/page.js` (Article detail)
  - `app/profile/page.js` (User profile)
  - `app/auth/login/page.js`
  - `app/auth/signup/page.js`
- [ ] Add navigation links in Header

Day 7: Development Tools

- [ ] Install dev dependencies:

```bash
npm install -D prettier eslint-config-prettier
```

- [ ] Create `.prettierrc` configuration
- [ ] Set up ESLint rules
- [ ] Create `utils/constants.js` for app-wide constants
- [ ] Create `utils/helpers.js` for common functions
- [ ] Test build: `npm run build`

🗄️ Phase 2: Database & Authentication (Week 2)

Day 8: Supabase Setup

- [ ] Create account at supabase.com
- [ ] Create new project named "grasp"

- [ ] Wait for project to initialize
- [ ] Copy project URL and anon key
- [ ] Add to `.env.local`:

```
NEXT_PUBLIC_SUPABASE_URL=your-project-url
NEXT_PUBLIC_SUPABASE_ANON_KEY=your-anon-key
```

Day 9: Database Schema Creation

- [ ] Go to Supabase SQL Editor
- [ ] Create `user_profiles` table
- [ ] Create `news_articles` table
- [ ] Create `user_article_interactions` table
- [ ] Create `personal_impacts` table
- [ ] Create `bias_analysis` table
- [ ] Create `api_usage_logs` table
- [ ] Add indexes for performance

Day 10: Supabase Configuration

- [ ] Create `lib/supabase.js` client configuration
- [ ] Set up Row Level Security (RLS) policies:
  - Users can only read/update their own profile
  - Articles are public read
  - Interactions are private to user
- [ ] Enable Realtime for news_articles table
- [ ] Test connection from your app

Day 11: Authentication Setup

- [ ] Enable Email Auth in Supabase
- [ ] Create `hooks/useAuth.js` custom hook
- [ ] Create login page UI
- [ ] Create signup page UI
- [ ] Implement login functionality
- [ ] Implement signup functionality
- [ ] Add logout button to Header

Day 12: Protected Routes

- [ ] Create `middleware.js` for route protection
- [ ] Protect `/profile` route
- [ ] Create auth context provider
- [ ] Wrap app with auth provider

- [ ] Test authentication flow
- [ ] Add "Remember me" functionality

Day 13: User Profile

- [ ] Create profile setup form
- [ ] Add fields: name, location, interests
- [ ] Create interests selection (checkboxes)
- [ ] Implement profile update functionality
- [ ] Create `api/user/profile` endpoint
- [ ] Test profile creation/update

Day 14: Authentication Polish

- [ ] Add loading states to auth forms
- [ ] Add error handling and display
- [ ] Create password reset flow
- [ ] Add social login buttons (UI only for now)
- [ ] Create welcome email template
- [ ] Test entire auth flow

📰 Phase 3: Core News Features (Week 3-4)

Day 15: API Keys Setup

- [ ] Sign up for NewsAPI.org
- [ ] Sign up for The Guardian API
- [ ] Sign up for Reddit API (create Reddit app)
- [ ] Add all API keys to `.env.local`
- [ ] Create `lib/apis/newsapi.js`
- [ ] Create `lib/apis/guardian.js`
- [ ] Create `lib/apis/reddit.js`

Day 16: News Fetching System

- [ ] Create `app/api/news/fetch/route.js`
- [ ] Implement NewsAPI fetching function
- [ ] Create data transformation function
- [ ] Add error handling
- [ ] Test API endpoint manually
- [ ] Create rate limiting logic

Day 17: Database Storage

- [ ] Create function to check if article exists

- [ ] Create function to store new articles
- [ ] Add duplicate detection (by URL)
- [ ] Implement batch insert for efficiency
- [ ] Create `utils/newsProcessor.js`
- [ ] Test storing articles in database

Day 18: Scheduled Fetching

- [ ] Create `app/api/cron/fetch-news/route.js`
- [ ] Set up Vercel Cron Jobs (vercel.json)
- [ ] Implement hourly fetching
- [ ] Add API usage tracking
- [ ] Create admin stats page
- [ ] Monitor API usage

Day 19: News Display Components

- [ ] Create `components/news/NewsCard.js`
- [ ] Create `components/news/NewsList.js`
- [ ] Create `components/news/NewsGrid.js`
- [ ] Add skeleton loading states
- [ ] Implement pagination
- [ ] Add category badges

Day 20: Article Detail Page

- [ ] Create article detail layout
- [ ] Add breadcrumb navigation
- [ ] Display full article content
- [ ] Add reading time calculator
- [ ] Create "Related Articles" section
- [ ] Add social share buttons

Day 21: Homepage Design

- [ ] Create hero section with tagline
- [ ] Add featured news carousel
- [ ] Create category sections
- [ ] Add "Latest News" feed
- [ ] Implement infinite scroll
- [ ] Add refresh button

🗺️ Phase 4: Interactive Map (Week 4)

Day 22: Map Setup

- [ ] Create `components/map/NewsMap.js`
- [ ] Initialize Leaflet map
- [ ] Add OpenStreetMap tiles
- [ ] Set default view (world view)
- [ ] Add zoom controls
- [ ] Fix Leaflet CSS issues in Next.js

Day 23: News Markers

- [ ] Create custom marker icons
- [ ] Add markers for news locations
- [ ] Implement marker clustering
- [ ] Add popup on marker click
- [ ] Show article preview in popup
- [ ] Link to full article

Day 24: Map Interactivity

- [ ] Add country boundaries layer
- [ ] Implement hover effects
- [ ] Create news density heatmap
- [ ] Add map controls (layers, zoom)
- [ ] Add "fly to" animation on country click
- [ ] Create map legend

Day 25: Location Services

- [ ] Sign up for IPInfo API
- [ ] Implement user location detection
- [ ] Add "Near me" button
- [ ] Filter news by proximity
- [ ] Create location-based recommendations
- [ ] Add location privacy settings

Day 26: Map Features

- [ ] Add sentiment color coding
- [ ] Create time-based filtering
- [ ] Add news category filtering on map
- [ ] Implement map search
- [ ] Add fullscreen mode
- [ ] Create mobile-optimized view

Day 27: Performance Optimization

- [ ] Implement marker virtualization
- [ ] Add progressive loading
- [ ] Optimize for mobile devices
- [ ] Cache map tiles
- [ ] Reduce marker re-renders
- [ ] Add loading indicators

Day 28: Week 4 Review

- [ ] Test all features built so far
- [ ] Fix any bugs found
- [ ] Optimize performance
- [ ] Update documentation
- [ ] Plan next phase
- [ ] Deploy to Vercel for testing

🤖 Phase 5: AI Integration (Week 5-6)

Day 29: OpenAI Setup

- [ ] Sign up for OpenAI API
- [ ] Add API key to environment
- [ ] Create `lib/apis/openai.js`
- [ ] Test API connection
- [ ] Implement error handling
- [ ] Set up usage monitoring

Day 30: Simplification Engine

- [ ] Create `app/api/ai/simplify/route.js`
- [ ] Design prompts for different levels:
  - ELI5 (Explain Like I'm 5)
  - Teen level
  - Standard simplification
- [ ] Implement caching for simplified content
- [ ] Add cost tracking

Day 31: Simplification UI

- [ ] Add "Simplify" button to articles
- [ ] Create reading level selector
- [ ] Add loading animation

- [ ] Display simplified content
- [ ] Add "Show original" toggle
- [ ] Save user's preferred level

Day 32: Summary Generation

- [ ] Create summary prompt template
- [ ] Generate 5-sentence summaries
- [ ] Add summary to article cards
- [ ] Create "Key points" extraction
- [ ] Add bullet point summaries
- [ ] Test with various article types

Day 33: Sentiment Analysis

- [ ] Sign up for Hugging Face
- [ ] Create `lib/apis/huggingface.js`
- [ ] Implement sentiment analysis
- [ ] Add sentiment badges to articles
- [ ] Create sentiment visualization
- [ ] Update map with sentiment colors

Day 34: Content Enhancement

- [ ] Create quiz generation for articles
- [ ] Add "Test your understanding" feature
- [ ] Generate discussion questions
- [ ] Create fact extraction
- [ ] Add keyword highlighting
- [ ] Implement read time estimates

Day 35: AI Features Polish

- [ ] Add retry logic for API failures
- [ ] Implement graceful degradation
- [ ] Create admin dashboard for AI usage
- [ ] Add user feedback on simplification
- [ ] Optimize API calls
- [ ] Test edge cases

🎯 Phase 6: Personalization (Week 7-8)

Day 36: Impact Analysis Design

- [ ] Design impact calculation algorithm

- [ ] Create impact categories:
  - Financial
  - Health
  - Work/Career
  - Travel
  - Lifestyle
- [ ] Design impact severity levels

Day 37: Impact Calculator

- [ ] Create `app/api/impact/calculate/route.js`
- [ ] Implement location-based impacts
- [ ] Add demographic considerations
- [ ] Create impact scoring system
- [ ] Test with various news types
- [ ] Store impacts in database

Day 38: Impact UI Components

- [ ] Create `components/ImpactCard.js`
- [ ] Design impact visualization
- [ ] Add severity indicators
- [ ] Create expandable details
- [ ] Add "Why this affects you" explanations
- [ ] Implement dismissible cards

Day 39: User Preferences

- [ ] Create preferences page
- [ ] Add interest categories selector
- [ ] Implement location settings
- [ ] Add notification preferences
- [ ] Create reading preferences
- [ ] Save to user profile

Day 40: Recommendation Engine

- [ ] Track user reading behavior
- [ ] Create interest scoring algorithm
- [ ] Implement collaborative filtering
- [ ] Add "Recommended for you" section
- [ ] Create "Discover" page
- [ ] Test recommendation accuracy

Day 41: Personalized Feed

- [ ] Create personalized homepage
- [ ] Implement feed algorithm
- [ ] Add interest-based sorting
- [ ] Create "For You" tab
- [ ] Add trending in your area
- [ ] Implement feed customization

Day 42: Engagement Features

- [ ] Add article bookmarking
- [ ] Create reading history
- [ ] Implement "Continue reading"
- [ ] Add reading streaks
- [ ] Create achievement system
- [ ] Design progress tracking

🔧 Phase 7: Advanced Features (Week 9-10)

Day 43: Bias Detection Setup

- [ ] Design bias detection logic
- [ ] Create source credibility database
- [ ] Implement multi-source fetching
- [ ] Create comparison algorithm
- [ ] Design bias indicators
- [ ] Set up bias categories

Day 44: Bias Analysis UI

- [ ] Create `components/BiasAnalysis.js`
- [ ] Design comparison view
- [ ] Add source badges
- [ ] Create bias spectrum visual
- [ ] Add "Facts vs Opinion" separator
- [ ] Implement source tooltips

Day 45: Social Features

- [ ] Add commenting system
- [ ] Create comment moderation
- [ ] Implement upvoting
- [ ] Add sharing functionality

- [ ] Create share tracking
- [ ] Design social cards

Day 46: Search Implementation

- [ ] Create search bar component
- [ ] Implement full-text search
- [ ] Add search filters
- [ ] Create search results page
- [ ] Add search suggestions
- [ ] Implement recent searches

Day 47: Analytics Dashboard

- [ ] Create user dashboard
- [ ] Add reading statistics
- [ ] Show interest evolution
- [ ] Create knowledge score
- [ ] Add category breakdowns
- [ ] Design data visualizations

Day 48: Performance Features

- [ ] Implement lazy loading
- [ ] Add image optimization
- [ ] Create service worker
- [ ] Enable offline reading
- [ ] Add PWA manifest
- [ ] Implement caching strategy

Day 49: Accessibility

- [ ] Add ARIA labels
- [ ] Implement keyboard navigation
- [ ] Create high contrast mode
- [ ] Add screen reader support
- [ ] Test with accessibility tools
- [ ] Fix accessibility issues

🚀 Phase 8: Polish & Launch (Week 11-12)

Day 50: UI Polish

- [ ] Refine all animations
- [ ] Add micro-interactions

- [ ] Polish loading states
- [ ] Create empty states
- [ ] Add error boundaries
- [ ] Implement toast notifications

Day 51: Mobile Optimization

- [ ] Test on various devices
- [ ] Fix responsive issues
- [ ] Optimize touch interactions
- [ ] Add pull-to-refresh
- [ ] Create app-like experience
- [ ] Test on slow connections

Day 52: SEO Optimization

- [ ] Add meta tags
- [ ] Create sitemap
- [ ] Implement Open Graph tags
- [ ] Add structured data
- [ ] Create robots.txt
- [ ] Optimize page titles

Day 53: Testing

- [ ] Create test plan
- [ ] Test all user flows
- [ ] Test edge cases
- [ ] Load testing
- [ ] Security testing
- [ ] Cross-browser testing

Day 54: Documentation

- [ ] Write user guide
- [ ] Create API documentation
- [ ] Document deployment process
- [ ] Create troubleshooting guide
- [ ] Write contributor guidelines
- [ ] Update README

Day 55: Beta Testing

- [ ] Deploy to production
- [ ] Invite beta testers

- [ ] Create feedback form
- [ ] Monitor error logs
- [ ] Track user behavior
- [ ] Collect feedback

Day 56: Bug Fixes

- [ ] Fix reported bugs
- [ ] Optimize based on feedback
- [ ] Improve performance issues
- [ ] Refine UI based on usage
- [ ] Update features
- [ ] Prepare for launch

Day 57: Launch Preparation

- [ ] Create landing page
- [ ] Write launch announcement
- [ ] Prepare social media posts
- [ ] Create demo video
- [ ] Set up analytics
- [ ] Final testing

Day 58: Launch! 🎉

- [ ] Deploy final version
- [ ] Announce on social media
- [ ] Submit to directories
- [ ] Monitor performance
- [ ] Respond to feedback
- [ ] Celebrate!

📝 Daily Routine Suggestion

Morning (1 hour)

- Review previous day's work
- Plan today's tasks
- Code implementation

Evening (1-2 hours)

- Continue coding
- Test features

- Commit changes
- Update task list

Weekend (2-3 hours)

- Catch up on missed tasks
- Plan next week
- Research new features
- Refactor code

🎯 Success Milestones

Week 2: Working authentication
Week 4: News displayed on map
Week 6: AI simplification working
Week 8: Personalized experience
Week 10: All features complete
Week 12: Launched! 🚀

💡 Pro Tips

1. **Commit daily**: Even small progress
2. **Test as you go**: Don't wait until the end
3. **Ask for help**: Join Next.js communities
4. **Stay motivated**: Your app will help millions understand news better
5. **Document everything**: Your future self will thank you