Project Implementation Plan

Technical Skills Learning & Practice Platform

Document Version: 1.0
Last Updated: November 19, 2025
Project Duration: 16-18 months (MVP in 3-4 months)
Methodology: Agile/Scrum with 2-week sprints
Team Size: 5-8 developers (recommended)

Executive Summary

This implementation plan provides a detailed, sprint-by-sprint roadmap for developing the Technical Skills Learning & Practice Platform. The plan follows Agile/Scrum methodology with 2-week sprints, organized into four major phases aligned with the PRD milestones[12][15].

Key Phases:

- **Phase 1 (MVP):** Sprints 1-8 (16 weeks / 4 months)
- **Phase 2 (Enhanced Features):** Sprints 9-14 (12 weeks / 3 months)
- **Phase 3 (Monetization):** Sprints 15-20 (12 weeks / 3 months)
- **Phase 4 (Advanced Features):** Sprints 21+ (Ongoing)

Development Methodology

Agile/Scrum Framework

Sprint Structure:

- **Duration:** 2 weeks per sprint[12][15]
- **Sprint Planning:** Monday Week 1 (2-4 hours)
- **Daily Standups:** 15 minutes every morning
- **Sprint Review:** Friday Week 2 (1-2 hours)
- **Sprint Retrospective:** Friday Week 2 (1 hour)
- **Sprint Velocity:** Track story points completed per sprint

Team Roles:

- **Product Owner:** Manages backlog, defines priorities
- **Scrum Master:** Facilitates ceremonies, removes blockers
- **Development Team:** 3-5 full-stack developers
- **DevOps Engineer:** Infrastructure and deployment
- **UI/UX Designer:** Interface design and user experience

Best Practices Applied:[12][15][18]

- Define clear sprint goals for each iteration
- Maintain balanced workloads to prevent burnout
- Allocate 20% capacity for unforeseen challenges
- Conduct thorough retrospectives after each sprint
- Use story points for estimation (Fibonacci: 1, 2, 3, 5, 8, 13)

Technology Stack Implementation Timeline

Development Environment Setup (Pre-Sprint 0)

Week -2 to Week 0:

Backend:

- Python 3.11+ with FastAPI framework
- PostgreSQL 15+ for primary database
- Redis 7+ for caching and sessions
- Docker and Docker Compose for containerization[14]
- SQLAlchemy ORM for database operations

Frontend:

- React 18+ with TypeScript
- Tailwind CSS for styling
- Monaco Editor for code editing
- Redux Toolkit for state management
- Axios for API communication

DevOps & Infrastructure:

- Git/GitHub for version control
- GitHub Actions for CI/CD pipeline[14]
- Docker for containerization
- AWS/DigitalOcean for cloud hosting (start small)
- Nginx as reverse proxy
- Prometheus + Grafana for monitoring (Phase 2)

Code Execution:

- Judge0 API (managed service for MVP)[13]
- Custom Docker executor (Phase 2 migration)
- RabbitMQ for async job queue

Development Tools:

- VS Code as primary IDE
- Postman for API testing
- pgAdmin for database management
- Figma for design collaboration
- Jira/Linear for project management

Phase 1: MVP Development (Sprints 1-8)

Sprint 0: Project Setup & Infrastructure (Week -2 to Week 0)

Sprint Goal: Establish development environment and project foundation

Tasks:

1. **Repository Setup**
   - Initialize Git repositories (backend, frontend)
   - Configure .gitignore and branch protection
   - Set up CI/CD pipeline skeleton
   - Story Points: 3
2. **Development Environment**
   - Docker Compose configuration for local development
   - Database initialization scripts
   - Environment variables management (.env templates)
   - Story Points: 5
3. **Project Structure**
   - Backend folder structure (controllers, models, services, routes)
   - Frontend folder structure (components, pages, utils, hooks)
   - API documentation setup (Swagger/OpenAPI)
   - Story Points: 3
4. **Database Schema v1**
   - Create initial migration scripts
   - Users, Courses, Topics, Problems tables
   - Indexes and constraints
   - Story Points: 5

Total Story Points: 16
Deliverable: Fully configured development environment

Sprint 1: User Authentication & Basic UI (Weeks 1-2)

Sprint Goal: Implement user registration, login, and basic navigation

Backend Tasks:

1. **User Model & Authentication**
   - User registration endpoint with email validation
   - Login endpoint with JWT token generation
   - Password hashing with bcrypt
   - Email verification service (SendGrid/Mailgun)
   - Story Points: 8
2. **Authorization Middleware**
   - JWT token validation middleware
   - Role-based access control (learner, admin)
   - Protected route decorators
   - Story Points: 5

Frontend Tasks:
3. **Authentication Pages**

- Registration form with validation
- Login form
- Email verification confirmation page
- Password reset request page
- Story Points: 8

4. **Basic Layout & Navigation**
    - Header with logo and navigation menu
    - Footer
    - Responsive sidebar (for logged-in users)
    - Public homepage skeleton
    - Story Points: 5

Total Story Points: 26
Deliverable: Working authentication system with basic UI

Sprint 2: Course Catalog & Course Detail (Weeks 3-4)

Sprint Goal: Create course browsing and enrollment functionality

Backend Tasks:

1. **Course Management API**
    - Create course CRUD endpoints
    - Get all courses with filtering (difficulty, tags)
    - Get single course with topics
    - Course enrollment endpoint
    - Story Points: 8
2. **Admin Course Management**
    - Admin-only course creation/editing endpoints
    - Course publish/unpublish functionality
    - Image upload for course thumbnails
    - Story Points: 5

Frontend Tasks:
3. **Course Catalog Page**

- Grid layout of course cards
- Filter by difficulty and tags
- Search functionality
- Pagination or infinite scroll
- Story Points: 8

4. **Course Detail Page**
    - Course overview and description
    - Topic index/table of contents
    - Enrollment button
    - Progress indicator (if enrolled)
    - Story Points: 5

Total Story Points: 26
Deliverable: Browse and enroll in courses

Sprint 3: Topic Content & Learning Interface (Weeks 5-6)

Sprint Goal: Build topic learning pages with rich content display

Backend Tasks:

1. **Topic Management API**
    - Get topic by ID with full content
    - Get topics by course ID
    - Topic CRUD for admins
    - Rich text content storage
    - Story Points: 5
2. **Content Rendering Service**
    - Markdown parsing and sanitization
    - Code syntax highlighting server-side
    - Image URL validation
    - Story Points: 3

Frontend Tasks:
3. **Topic Learning Page**

- Markdown content rendering
- Code block syntax highlighting
- Image and video embedding
- Next/Previous topic navigation
- Table of contents sidebar
- Story Points: 8

4. **Progress Tracking UI**
    - Mark topic as completed checkbox
    - Progress bar for course completion
    - Backend API for progress updates
    - Story Points: 5

Total Story Points: 21
Deliverable: Complete topic learning experience

Sprint 4: Problem Management & Display (Weeks 7-8)

Sprint Goal: Create problem creation and display functionality

Backend Tasks:

1. **Problem Model & API**
    - Problem CRUD endpoints
    - Get problems by topic ID
    - Problem difficulty and tags
    - Sample test cases storage
    - Story Points: 8
2. **Admin Problem Editor API**
    - Create/edit problem statements (Markdown)
    - Add/edit test cases (input/output pairs)
    - Set time and memory limits

- Problem versioning
- Story Points: 8

Frontend Tasks:
3. **Problem List Page**

- Display problems for a topic
- Filter by difficulty
- Problem status indicators (solved/unsolved)
- Story Points: 5

4. **Problem Statement Page**
   - Problem description rendering
   - Sample input/output display
   - Constraints and notes
   - Link to code editor
   - Story Points: 5

Total Story Points: 26
Deliverable: Problem management system

Sprint 5: Code Editor Integration (Weeks 9-10)

Sprint Goal: Integrate Monaco Editor with language selection

Frontend Tasks:

1. **Monaco Editor Setup**
   - Install and configure Monaco Editor
   - Language selector dropdown (Python, Java, C++, JavaScript)
   - Theme toggle (Light/Dark)
   - Code templates for each language
   - Story Points: 8
2. **Editor UI/UX**
   - Split view: problem statement + editor
   - Resizable panels
   - Font size controls
   - Code formatting button
   - Reset code button
   - Story Points: 5
3. **Local Code Testing**
   - Input/output test area
   - "Run Code" button
   - Display output/errors
   - Sample test case validation
   - Story Points: 5

Backend Tasks:
4. **Code Storage API**

- Save user's code drafts
- Retrieve last saved code for a problem

- Auto-save mechanism (every 30 seconds)
- Story Points: 5

Total Story Points: 23
Deliverable: Functional code editor

Sprint 6: Code Execution Engine (Weeks 11-12)

Sprint Goal: Implement sandboxed code execution and evaluation

Backend Tasks:

1. **Judge0 Integration**
    - Set up Judge0 API account (or self-hosted)
    - Create submission service
    - Language ID mapping (Python=71, Java=62, C++=54, etc.)
    - Story Points: 8
2. **Code Execution Service**
    - Submit code to Judge0
    - Poll for results or use webhook
    - Parse execution results
    - Handle errors (compilation, runtime, timeout)
    - Story Points: 8
3. **Test Case Evaluation**
    - Run code against all test cases
    - Compare outputs (exact match, whitespace handling)
    - Calculate final verdict (AC/WA/TLE/MLE/RE)
    - Store submission results
    - Story Points: 8
4. **Submission API**
    - Create submission endpoint
    - Get submission results
    - Get submission history for user+problem
    - Story Points: 5

Frontend Tasks:
5. **Submission UI**

- "Submit" button
- Loading state during evaluation
- Results display (verdict, time, memory)
- Test case results (passed/failed)
- Error messages display
- Story Points: 5

Total Story Points: 34 (High complexity sprint)
Deliverable: End-to-end code execution

Sprint 7: User Dashboard & Progress Tracking (Weeks 13-14)

Sprint Goal: Build user dashboard with progress metrics

Backend Tasks:

1. **User Progress API**
    - Get user's enrolled courses
    - Calculate completion percentages
    - Get problems solved count
    - Get submission statistics
    - Story Points: 8
2. **Dashboard Analytics Service**
    - Total problems solved by difficulty
    - Languages used statistics
    - Streak calculation (consecutive days)
    - Recent activity feed
    - Story Points: 5

Frontend Tasks:
3. **Dashboard Page**

- Welcome section with user stats
- Enrolled courses cards
- Progress bars and charts
- Recent submissions list
- Quick access to continue learning
- Story Points: 8

4. **Profile Page**
    - Display user information
    - Edit profile form
    - Activity heatmap (GitHub-style)
    - Problems solved breakdown
    - Story Points: 8

Total Story Points: 29
Deliverable: User dashboard and profile

Sprint 8: Admin Panel & MVP Polish (Weeks 15-16)

Sprint Goal: Complete admin dashboard and prepare for MVP launch

Backend Tasks:

1. **Admin Analytics API**
    - Platform statistics (users, courses, submissions)
    - Usage trends over time
    - Popular courses and problems
    - Story Points: 5

Frontend Tasks:
2. **Admin Dashboard**

- Platform overview with key metrics
- User management table
- Course management interface
- Problem management interface
- Story Points: 13

3. **Content Creation Workflows**
    - Rich text editor for course content
    - Problem statement editor with preview
    - Test case bulk upload (CSV/JSON)
    - Story Points: 8
4. **MVP Bug Fixes & Polish**
    - Fix critical bugs from testing
    - Performance optimization
    - Responsive design fixes
    - Accessibility improvements
    - Story Points: 8
5. **Documentation**
    - User guide/FAQ
    - Admin documentation
    - API documentation update
    - Story Points: 3

Total Story Points: 37
Deliverable: MVP ready for beta launch

MVP Launch Checklist

Pre-Launch (Week 16):

- [ ] Load testing (simulate 500+ concurrent users)
- [ ] Security audit (OWASP Top 10 checks)
- [ ] Database backup strategy implemented
- [ ] Monitoring and alerting configured
- [ ] SSL certificate installed
- [ ] DNS configuration complete
- [ ] Privacy policy and terms of service published
- [ ] Email templates tested
- [ ] Error tracking (Sentry) configured

Beta Launch (Week 17-20):

- Invite 50-100 beta users
- Collect feedback through surveys
- Monitor error logs and performance
- Fix high-priority issues
- Prepare marketing materials

MVP Success Criteria:

- 500+ registered users
- 1,000+ problem submissions

- <3 second average page load time
- 99% uptime
- <5% error rate
- 60% user retention after 7 days

Phase 2: Enhanced Features (Sprints 9-14)

Sprint 9: Discussion Forum Foundation (Weeks 17-18)

Sprint Goal: Build discussion forum infrastructure

Backend Tasks:

1. **Discussion Model & API**
    - Discussion thread CRUD
    - Comment/reply system (nested threads)
    - Upvote/downvote functionality
    - Thread association (problem/course/general)
    - Story Points: 13
2. **Discussion Service**
    - Pagination for threads and comments
    - Sort by latest/most upvoted
    - Search discussions
    - Story Points: 5

Frontend Tasks:
3. **Discussion Forum UI**

- Thread list page with filters
- Create new discussion form
- Thread detail page with comments
- Reply and nested reply components
- Story Points: 13

Total Story Points: 31
Deliverable: Basic discussion forum

Sprint 10: Solution Sharing System (Weeks 19-20)

Sprint Goal: Enable users to share and view solutions

Backend Tasks:

1. **Solution Model & API**
    - Solution submission endpoint
    - Get solutions for a problem
    - Solution access control (locked/unlocked)
    - Upvote/downvote solutions
    - Story Points: 8
2. **Solution Unlock Logic**
    - Unlock after solving problem
    - Unlock after N failed attempts (configurable)

    - Premium access (future)
    - Story Points: 5

Frontend Tasks:
3. **Solution Sharing UI**

- Submit solution form with explanation
- Solution list page
- Solution detail view with code highlighting
- Lock/unlock indicators
- Comment on solutions
- Story Points: 8

4. **Solution Integration**
    - "View Solutions" button on problem page
    - Solution count indicator
    - Best solution highlighting
    - Story Points: 3

Total Story Points: 24
Deliverable: Solution sharing platform

Sprint 11: Challenge System - Daily Challenges (Weeks 21-22)

Sprint Goal: Implement daily challenge functionality

Backend Tasks:

1. **Challenge Model & API**
    - Challenge creation and scheduling
    - Challenge types (daily/weekly/monthly)
    - Challenge-problem association
    - Challenge status management
    - Story Points: 8
2. **Daily Challenge Service**
    - Auto-publish daily challenge at midnight UTC
    - Challenge participation tracking
    - Calculate time-based scoring
    - Daily leaderboard generation
    - Story Points: 13

Frontend Tasks:
3. **Challenge Pages**

- Challenges landing page
- Daily challenge card
- Challenge problem page
- Countdown timer to next challenge
- Story Points: 8

4. **Challenge Participation**
    - "Start Challenge" button
    - Submission within challenge context

- Real-time score updates
- Challenge completion badge
- Story Points: 5

Total Story Points: 34
Deliverable: Daily challenges live

Sprint 12: Challenge System - Contests & Leaderboards (Weeks 23-24)

Sprint Goal: Add weekly/monthly contests with leaderboards

Backend Tasks:

1. **Contest Management**
   - Weekly contest scheduling
   - Multiple problems per contest
   - Contest duration enforcement
   - Penalty for wrong submissions
   - Story Points: 8
2. **Leaderboard Service**
   - Real-time ranking calculation
   - Redis for leaderboard caching
   - Historical leaderboard storage
   - Filter by country/region
   - Story Points: 8

Frontend Tasks:
3. **Contest Interface**

- Active contest page
- Problem list with scores
- Contest timer
- Live leaderboard with auto-refresh
- Story Points: 13

4. **Contest History**
   - Past contests page
   - User's contest participation history
   - Virtual contest mode (practice past contests)
   - Story Points: 5

Total Story Points: 34
Deliverable: Full contest system

Sprint 13: Gamification & Achievements (Weeks 25-26)

Sprint Goal: Implement badges, streaks, and levels

Backend Tasks:

1. **Achievement System**
   - Badge definitions and triggers
   - Achievement unlock service

    - Streak calculation (daily activity)
    - Level/rank calculation
    - Story Points: 13
2. **Achievement API**
    - Get user achievements
    - Badge notification system
    - Leaderboard for achievements
    - Story Points: 5

Frontend Tasks:
3. **Achievement UI**

- Badges display on profile
- Achievement unlock notifications
- Streak counter on dashboard
- Level/rank badge
- Achievements showcase page
- Story Points: 8

4. **Enhanced Dashboard**
    - Activity heatmap (GitHub-style)
    - Streak visualization
    - Progress towards next level
    - Story Points: 5

Total Story Points: 31
Deliverable: Gamification system

Sprint 14: Social Features & Notifications (Weeks 27-28)

Sprint Goal: Add social auth, notifications, and sharing

Backend Tasks:

1. **OAuth Integration**
    - Google OAuth setup
    - GitHub OAuth setup
    - Link social accounts to existing users
    - Story Points: 8
2. **Notification Service**
    - Email notification templates
    - Notification preferences management
    - Challenge reminder emails
    - Achievement unlock emails
    - Story Points: 8

Frontend Tasks:
3. **Social Login Buttons**

- Add OAuth buttons to login/signup
- Profile social connections
- Story Points: 3

4. **Notification Center**
    - In-app notification bell icon
    - Notification list dropdown
    - Mark as read functionality
    - Notification preferences page
    - Story Points: 8
5. **Sharing Features**
    - Share profile button
    - Share achievement on social media
    - Embed code for profile widget
    - Story Points: 5

Total Story Points: 32
Deliverable: Social features complete

Phase 3: Monetization & Scale (Sprints 15-20)

Sprint 15: Premium Tier Infrastructure (Weeks 29-30)

Sprint Goal: Build subscription management foundation

Backend Tasks:

1. **Subscription Model**
    - User subscription table
    - Plan definitions (free/premium)
    - Subscription status tracking
    - Trial period logic
    - Story Points: 8
2. **Payment Integration (Stripe)**
    - Stripe account setup
    - Checkout session creation
    - Webhook handling for payment events
    - Subscription renewal automation
    - Story Points: 13
3. **Access Control Updates**
    - Premium-only content flagging
    - Middleware for premium features
    - Upgrade prompts for free users
    - Story Points: 5

Frontend Tasks:
4. **Pricing Page**

- Plan comparison table
- Feature breakdown (free vs premium)
- Checkout flow
- Story Points: 8

Total Story Points: 34
Deliverable: Subscription system

Sprint 16: Premium Features (Weeks 31-32)

Sprint Goal: Implement premium-exclusive features

Backend Tasks:

1. **Premium Content Service**
    - Premium-only courses and challenges
    - Unlimited solution access for premium users
    - Priority code execution queue
    - Story Points: 8
2. **Certificate Generation**
    - Course completion certificate creation
    - PDF generation with user details
    - Certificate verification system
    - Story Points: 8

Frontend Tasks:
3. **Premium Dashboard**

- Premium user badge
- Download certificates
- Premium-only content section
- Ad-free experience
- Story Points: 5

4. **Upgrade Prompts**
    - Locked solution unlock with upgrade CTA
    - Premium challenge banners
    - Limited feature tooltips
    - Story Points: 5

Total Story Points: 26
Deliverable: Premium features live

Sprint 17: Skills Profile & Portfolio (Weeks 33-34)

Sprint Goal: Build shareable skills profile for job seekers

Backend Tasks:

1. **Public Profile API**
    - Public profile endpoint
    - Shareable profile URL (username-based)
    - Privacy settings management
    - Story Points: 8
2. **Profile Analytics**
    - Skills assessment based on solved problems
    - Topic expertise calculation
    - Challenge ranking highlights
    - Story Points: 8

Frontend Tasks:
3. **Public Profile Page**

- Professional profile layout
- Skills and expertise tags
- Activity heatmap
- Top solutions showcase
- Challenge achievements
- Story Points: 13

4. **Profile Customization**
    - Edit bio and links
    - Choose visible sections
    - Profile theme customization
    - Download profile as PDF
    - Story Points: 8

Total Story Points: 37
Deliverable: Shareable skills profile

Sprint 18: Mobile Responsiveness & PWA (Weeks 35-36)

Sprint Goal: Optimize for mobile and add PWA capabilities

Frontend Tasks:

1. **Mobile UI Optimization**
    - Responsive code editor for mobile
    - Touch-friendly navigation
    - Mobile-optimized problem solving
    - Story Points: 13
2. **PWA Implementation**
    - Service worker setup
    - Offline caching strategy
    - Add to home screen functionality
    - Push notification support
    - Story Points: 13
3. **Performance Optimization**
    - Code splitting and lazy loading
    - Image optimization
    - Bundle size reduction
    - Story Points: 8

Total Story Points: 34
Deliverable: Mobile-optimized PWA

Sprint 19: Analytics & Monitoring (Weeks 37-38)

Sprint Goal: Implement comprehensive analytics and monitoring

Backend Tasks:

1. **Analytics Service**
    - Google Analytics integration
    - Custom event tracking
    - User behavior analytics
    - Conversion funnel tracking
    - Story Points: 8
2. **Monitoring Stack**
    - Prometheus metrics collection
    - Grafana dashboards
    - Alert rules configuration
    - Story Points: 8
3. **Logging Infrastructure**
    - Centralized logging (ELK stack or similar)
    - Log aggregation from all services
    - Error tracking with Sentry
    - Story Points: 8

Frontend Tasks:
4. **Admin Analytics Dashboard**

- User growth charts
- Engagement metrics
- Revenue analytics
- Retention cohort analysis
- Story Points: 8

Total Story Points: 32
Deliverable: Full analytics platform

Sprint 20: Performance & Security Hardening (Weeks 39-40)

Sprint Goal: Optimize performance and security for scale

Backend Tasks:

1. **Performance Optimization**
    - Database query optimization
    - Redis caching strategy expansion
    - API response time improvements
    - CDN setup for static assets
    - Story Points: 13
2. **Security Enhancements**
    - Security audit and penetration testing
    - Rate limiting on all endpoints
    - Input validation improvements
    - OWASP Top 10 compliance

- Two-factor authentication
- Story Points: 13

3. **Scalability Preparation**
   - Database connection pooling
   - Load balancer configuration
   - Horizontal scaling readiness
   - Story Points: 8

Total Story Points: 34
Deliverable: Production-hardened platform

Phase 4: Advanced Features (Sprints 21+)

Sprint 21: Advanced Code Execution (Weeks 41-42)

Sprint Goal: Migrate to custom code execution infrastructure

Backend Tasks:

1. **Custom Docker Executor**
   - Build Docker images for each language
   - Implement resource limiting
   - Security sandboxing
   - Story Points: 21
2. **Execution Queue Optimization**
   - RabbitMQ/Celery queue setup
   - Worker scaling logic
   - Priority queue for premium users
   - Story Points: 8

Total Story Points: 29
Deliverable: Custom execution engine

Sprint 22-23: AI-Powered Features (Weeks 43-46)

Sprint Goal: Integrate AI for hints and recommendations

Backend Tasks:

1. **AI Service Integration**
   - OpenAI API integration
   - Problem hint generation
   - Code explanation service
   - Story Points: 13
2. **Recommendation Engine**
   - Problem recommendation based on history
   - Course recommendation
   - Learning path suggestions
   - Story Points: 13

Frontend Tasks:
3. **AI Features UI**

- Hint request button
- Code explanation tooltip
- Recommended problems section
- Story Points: 8

Total Story Points: 34 (2 sprints)
Deliverable: AI-enhanced learning

Sprint 24-25: Video Courses & Tutorials (Weeks 47-50)

Sprint Goal: Add video content support

Backend Tasks:

1. **Video Integration**
   - Video upload to cloud storage
   - Video streaming setup
   - Video metadata management
   - Story Points: 13
2. **Video Progress Tracking**
   - Track video watch time
   - Mark video as completed
   - Resume from last position
   - Story Points: 8

Frontend Tasks:
3. **Video Player**

- Custom video player component
- Playback speed controls
- Transcript display
- Story Points: 8

4. **Video Course Pages**
   - Video-based course structure
   - Video library organization
   - Story Points: 5

Total Story Points: 34 (2 sprints)
Deliverable: Video learning platform

Ongoing: Continuous Improvement

Every Sprint (Post-MVP):

- User feedback review and implementation (3-5 story points)
- Bug fixes from production (5-8 story points)
- Content addition (new courses, problems) (3-5 story points)
- Performance monitoring and optimization (2-3 story points)
- Security updates and patches (2-3 story points)

Resource Planning

Team Structure (MVP Phase)

Core Team (5-6 members):

- 1 Product Owner (part-time, 20hrs/week)
- 1 Scrum Master (can be combined with developer)
- 2-3 Full-Stack Developers (Python + React)
- 1 DevOps Engineer (part-time, 20hrs/week)
- 1 UI/UX Designer (contract, as needed)

Extended Team (Phase 2+):

- Additional 1-2 developers
- QA Engineer
- Content Creator (technical writer)
- Marketing/Growth specialist

Budget Estimation (MVP - 4 months)

Personnel (Primary cost):

- Full-Stack Developers: 3 × ₹50,000/month × 4 = ₹6,00,000
- DevOps Engineer: ₹25,000/month × 4 = ₹1,00,000
- UI/UX Designer: ₹40,000 (contract)
- **Subtotal: ₹7,40,000**

Infrastructure & Tools:

- Cloud Hosting (AWS/DigitalOcean): ₹10,000/month × 4 = ₹40,000
- Judge0 API: ₹5,000/month × 4 = ₹20,000
- Domain & SSL: ₹2,000
- Third-party services (SendGrid, etc.): ₹5,000/month × 4 = ₹20,000
- Development tools (licenses): ₹10,000
- **Subtotal: ₹92,000**

Miscellaneous:

- Buffer (15% of total): ₹1,25,000
- **Total MVP Budget: ₹9,57,000 (~₹10,00,000)**

Infrastructure Cost Scaling

MVP (500 users): ₹10,000/month
Phase 2 (5,000 users): ₹30,000/month
Phase 3 (20,000 users): ₹80,000/month
Phase 4 (100,000 users): ₹3,00,000/month

Risk Management

Technical Risks

Risk	Probability	Impact	Mitigation Strategy
Code execution security breach	Medium	High	Use Judge0 for MVP, implement strict sandboxing in Phase 2, regular security audits
Database performance issues	High	Medium	Implement caching early, use read replicas, optimize queries from Sprint 1
Scalability bottlenecks	Medium	High	Design for horizontal scaling, use microservices patterns, load test before each phase launch
Third-party API downtime (Judge0)	Medium	High	Have fallback execution engine ready, implement queue retry logic, SLA monitoring
Team velocity lower than estimated	High	Medium	Buffer 20% in sprint planning, prioritize ruthlessly, consider hiring contractors

Business Risks

Risk	Probability	Impact	Mitigation Strategy
Low user adoption	Medium	High	Beta testing with target users, SEO optimization, content marketing, referral program
High user churn	Medium	High	Implement gamification early, daily challenges for engagement, user feedback loops
Funding shortage	Medium	High	Secure funding before Phase 3, consider bootstrapping MVP, explore grants/competitions
Competition intensifies	High	Medium	Focus on unique features (India-centric content, better UX), build community, fast iteration

Timeline Risks

Risk	Probability	Impact	Mitigation Strategy
Sprint delays accumulate	High	Medium	Weekly sprint reviews, identify blockers early, adjust scope not deadline
Key developer leaves	Low	High	Knowledge documentation, pair programming, backup resources identified
Scope creep	High	Medium	Strict prioritization, product owner approval required, "nice-to-have" parking lot

---

Quality Assurance Strategy

Testing Pyramid

Unit Tests (60% of tests):

- Backend: pytest for all services and utilities
- Frontend: Jest for React components and utilities
- Target: 80% code coverage
- Run on every commit via CI/CD

Integration Tests (30% of tests):

- API endpoint testing with test database
- Code execution workflow end-to-end
- Authentication and authorization flows
- Run on every pull request

E2E Tests (10% of tests):

- Cypress for critical user journeys
- User registration and login
- Problem solving workflow
- Challenge participation
- Run nightly and before releases

Code Review Process

Every Pull Request Must Have:

- At least one approval from another developer
- Passing CI/CD pipeline (tests + linting)
- No merge conflicts
- Updated documentation if needed

Review Checklist:

- Code follows style guide (PEP 8 for Python, ESLint for JS)
- Proper error handling
- Security considerations addressed
- Performance implications considered
- Tests included for new features

Release Process

Staging Environment:

- Deploy to staging after sprint review
- QA testing on staging (1-2 days)
- User acceptance testing (if applicable)

Production Deployment:

- Deploy during low-traffic hours
- Database migrations run first
- Feature flags for risky features

- Rollback plan ready
- Monitor for 2 hours post-deployment

Success Metrics & KPIs

Sprint-Level Metrics

Velocity Tracking:

- Story points committed vs completed
- Sprint burndown charts
- Velocity trend over time
- Target: Stabilize velocity by Sprint 3

Quality Metrics:

- Bugs found in sprint vs bugs carried over
- Code review turnaround time (<24 hours)
- Test coverage percentage (maintain >75%)
- Build success rate (>95%)

Phase-Level Goals

MVP (End of Sprint 8):

- ✓ 500+ registered users
- ✓ 50+ courses/topics created
- ✓ 100+ practice problems
- ✓ 1,000+ code submissions
- ✓ 60% Day-7 retention
- ✓ <2 second average page load
- ✓ 99% uptime

Phase 2 (End of Sprint 14):

- ✓ 5,000+ registered users
- ✓ 500+ daily active users
- ✓ 10,000+ monthly submissions
- ✓ 100+ daily challenge participants
- ✓ 50+ discussion threads weekly
- ✓ 70% Day-30 retention

Phase 3 (End of Sprint 20):

- ✓ 20,000+ registered users
- ✓ 2,000+ daily active users
- ✓ 5% premium conversion rate
- ✓ ₹3,00,000+ monthly recurring revenue
- ✓ 80% user satisfaction (NPS >40)
- ✓ 50,000+ monthly submissions

Communication & Collaboration

Daily Standup Format (15 minutes)

Each team member shares:

1. What I completed yesterday
2. What I'm working on today
3. Any blockers or help needed

Scrum Master responsibilities:

- Time-box to 15 minutes
- Note blockers and action items
- Schedule deeper discussions offline

Sprint Ceremonies

Sprint Planning (Monday Week 1, 2-4 hours):

- Product Owner presents sprint goal
- Team reviews and estimates backlog items
- Commitment to sprint backlog
- Task breakdown and assignment

Sprint Review (Friday Week 2, 1-2 hours):

- Demo completed features to stakeholders
- Gather feedback
- Update product backlog based on feedback

Sprint Retrospective (Friday Week 2, 1 hour):

- What went well?
- What could be improved?
- Action items for next sprint
- Process adjustments

Communication Channels

Synchronous:

- Daily standups (Zoom/Google Meet)
- Slack for quick questions and updates
- Pair programming sessions (as needed)

Asynchronous:

- Jira/Linear for task management
- GitHub for code reviews and discussions
- Confluence/Notion for documentation
- Weekly sprint status email

Knowledge Management

Documentation Requirements

Technical Documentation:

- API documentation (Swagger/OpenAPI) - updated every sprint
- Architecture decision records (ADRs)
- Database schema documentation
- Deployment runbooks
- Troubleshooting guides

User Documentation:

- User guide for learners
- Admin manual
- FAQ section
- Video tutorials (Phase 2+)

Team Documentation:

- Onboarding guide for new developers
- Coding standards and style guide
- Git workflow and branching strategy
- Testing guidelines
- Security best practices

Knowledge Sharing

Weekly:

- Tech talks on new technologies (30 minutes)
- Code review learnings shared

Monthly:

- Lunch and learn sessions
- External speaker/webinar

Continuous:

- Pair programming rotations
- Code comments and documentation
- Team wiki updates

Post-Launch Strategy

Beta Testing (Weeks 17-20)

Beta User Recruitment:

- Invite computer science students (50 users)
- Reach out to coding bootcamps (30 users)
- Personal network and social media (20 users)

Feedback Collection:

- In-app feedback widget
- User interview sessions (10-15 users)
- Survey after 2 weeks of usage
- Analytics review (user behavior)

Iteration:

- Prioritize top 10 pain points
- Quick fixes in Sprints 9-10
- Major improvements in Phase 2 roadmap

Marketing & Growth

Pre-Launch (Week 15-16):

- Build landing page with email signup
- Share on LinkedIn, Twitter, Reddit (r/learnprogramming, r/cscareerquestions)
- Reach out to college coding clubs

Launch (Week 17):

- Product Hunt launch
- LinkedIn article about the platform
- Email beta waitlist
- Free courses to first 100 users

Post-Launch (Ongoing):

- SEO optimization (target keywords: "learn DSA", "coding practice", "programming challenges")
- Content marketing (blog posts on interview prep)
- YouTube channel with problem walkthroughs
- Referral program (Phase 2)
- Partnerships with colleges and bootcamps

Community Building

Early Community:

- Discord server for users
- Weekly problem-solving sessions
- Monthly contests with prizes
- Featured user showcase

Content Strategy:

- 2-3 new courses per month (Phase 2+)
- Daily challenge problems
- Weekly editorials and solution videos
- Interview experiences and tips

Appendix: Sprint Planning Templates

Sprint Planning Template

Sprint Number: [X]
Sprint Goal: [One sentence goal]
Duration: [Start Date] - [End Date]
Team Capacity: [Total story points available]

Backlog Items:

Task	Story Points	Assignee	Priority
[Task 1]	[5]	[Name]	High
[Task 2]	[8]	[Name]	High
[Task 3]	[3]	[Name]	Medium

Total Committed: [XX] story points

Definition of Done:

- [ ] Code complete and reviewed
- [ ] Unit tests written and passing
- [ ] Integration tests passing
- [ ] Documentation updated
- [ ] Deployed to staging
- [ ] QA tested
- [ ] Product owner approved

Story Point Reference Guide

Points	Complexity	Time Estimate	Example
1	Trivial	1-2 hours	Fix typo, update copy
2	Simple	2-4 hours	Add new API endpoint with basic logic
3	Easy	4-8 hours	Create simple UI component
5	Medium	1-2 days	Implement user authentication
8	Complex	2-4 days	Build code execution service
13	Very Complex	4-7 days	Implement challenge system
21	Extremely Complex	1-2 weeks	Custom Docker executor (consider splitting)

Note: Tasks >13 points should be broken down into smaller tasks

---

Conclusion

This implementation plan provides a comprehensive, sprint-by-sprint roadmap for building the Technical Skills Learning & Practice Platform. The plan balances speed (MVP in 4 months) with quality and scalability[13][19].

Key Success Factors:

1. **Disciplined Agile execution:** Stick to 2-week sprints and retrospectives[12][15]
2. **Focused MVP:** Resist scope creep, launch early with core features
3. **User-centric development:** Beta testing and feedback loops[18]
4. **Technical excellence:** Code reviews, testing, documentation from Day 1
5. **Scalability mindset:** Build for growth using microservices patterns[14][17]
6. **Continuous iteration:** Use data and feedback to guide priorities

Next Steps:

1. Secure funding/team commitment
2. Set up development environment (Sprint 0)
3. Hire core team members
4. Begin Sprint 1 with user authentication
5. Ship MVP in 16 weeks
6. Iterate based on user feedback

The platform has strong potential to compete with established players by focusing on user experience, community building, and localized content. With disciplined execution and user-focused iteration, this implementation plan sets the foundation for a successful educational platform.

Document Approval:

Role	Name	Signature	Date
Product Owner	_____	_____	_____
Tech Lead	_____	_____	_____
Scrum Master	_____	_____	_____

References:

[12] DevDynamics - The Ultimate Guide to Agile Sprint Planning (2024)
[13] KVY Technology - Software Development Timeline Estimation (2024)
[14] Roadmap.sh - Scalable E-Commerce Platform with Microservices (2025)
[15] Shorepod - 9 Agile Software Development Best Practices for 2025
[16] Designveloper - 10 Stages in App Development Timeline (2025)
[17] GeeksforGeeks - What are Microservices? (2023)
[18] Umano - Sprint Planning Best Practices (2025)
[19] Soltech - Decoding Software Development Timeline (2025)

End of Implementation Plan