

# LINEAR - Product Requirements Document (PRD)

**Versão:** 1.0

**Data:** Janeiro 2026

**Desenvolvido por:** Senior Developer & UX/UI Designer

**Status:** Produção

---

## ▮ Sumário Executivo

Linear é uma ferramenta de **rastreamento de problemas e gerenciamento de projetos** otimizada para equipes de produto e engenharia de alto desempenho. Diferentemente de softwares tradicionais que tentam fazer tudo para todos, Linear foca em **velocidade, simplicidade e clareza de fluxo**.

### Objetivo Principal

Criar um sistema intuitivo que permite que equipes de desenvolvimento rastreiem, priorizem e resolvam tarefas com máxima eficiência, reduzindo overhead administrativo e mantendo comunicação centralizada.

### Público-alvo

- Equipes de engenharia (2-500+ pessoas)
  - Product managers e líderes de produto
  - Designers e PMs técnicos
  - Equipes de Customer Support
  - Startups e empresas SaaS
- 

## ▮ Objetivos e Escopo

### Objetivos Principais

1. **Velocidade:** Criar issues em menos de 3 segundos com atalhos de teclado
2. **Simplicidade:** Interface limpa, sem cluttering visual
3. **Automação:** Workflows automáticos para reduzir tarefas repetitivas
4. **Colaboração:** Real-time collaboration e mentions inteligentes
5. **Integração:** Sincronização perfeita com GitHub, Slack, Figma, Sentry

### Escopo v1.0

- ✓ Rastreamento de issues (Backlog, In Progress, Done, etc.)
- ✓ Sistema de ciclos (sprints)
- ✓ Roadmap e projetos
- ✓ Integração com GitHub e Slack
- ✓ Automações básicas

- ✓ Temas personalizados (light/dark)

## Out of Scope (v2.0+)

- Colaboração em tempo real em documentos (v2.0)
  - Integração com Jira (v2.1)
  - Relatórios avançados e BI (v2.0)
  - Mobile app nativo (v2.0)
- 

## □ Persona e Casos de Uso

### Persona 1: Marco (Senior Engineer)

**Objetivo:** Rastrear bugs e features com máxima velocidade

**Necessidade:** Criar issues em segundos, vincular PRs do GitHub automaticamente, mover tarefas rapidamente

**Pain Point:** Ferramentas lentas que interrompem fluxo de trabalho

**Caso de Uso:**

Marco cria uma issue pelo Linear CLI em 2 segundos

- Issue é linkada automaticamente a seu PR no GitHub
- Quando PR é merged, status muda para Done automaticamente
- Team é notificado no Slack

### Persona 2: Sofia (Product Manager)

**Objetivo:** Gerenciar roadmap, prioridades e progresso de 20+ projetos

**Necessidade:** Visão holística, relatórios de progresso, planejamento de sprints

**Pain Point:** Falta de visibilidade sobre progress real vs. planejado

**Caso de Uso:**

Sofia define Q1 goals no Linear

- Cria projects com milestones
- Atribui issues a diferentes squads
- Acompanha velocity e atraso em tempo real
- Exporta roadmap para stakeholders

### Persona 3: Alex (Designer)

**Objetivo:** Colaborar com engineers, fornecer specs, rastrear implementação

**Necessidade:** Compartilhar designs, vincular Figma links, ver status de implementação

**Pain Point:** Silo entre design e eng, falta de contexto

**Caso de Uso:**

Alex cria issue para nova feature

- Adiciona link de design do Figma
  - Marco vê design junto com specs no Linear
  - Alex pode comentar em progresso
  - Integração mostra quando design foi implementado
-

# ▯ Arquitetura do Sistema

## Stack Tecnológico

### Frontend:

- React 18+ (com TypeScript)
- Next.js 14+ para SSR/SSG
- Tailwind CSS + Design System customizado
- TanStack Query para state management
- WebSocket para real-time updates
- Electron para desktop app (macOS/Windows)

### Backend:

- Node.js/Express ou Deno
- PostgreSQL para dados primários
- Redis para cache e sessions
- GraphQL API (Apollo Server)
- WebSocket server ([Socket.io](#))

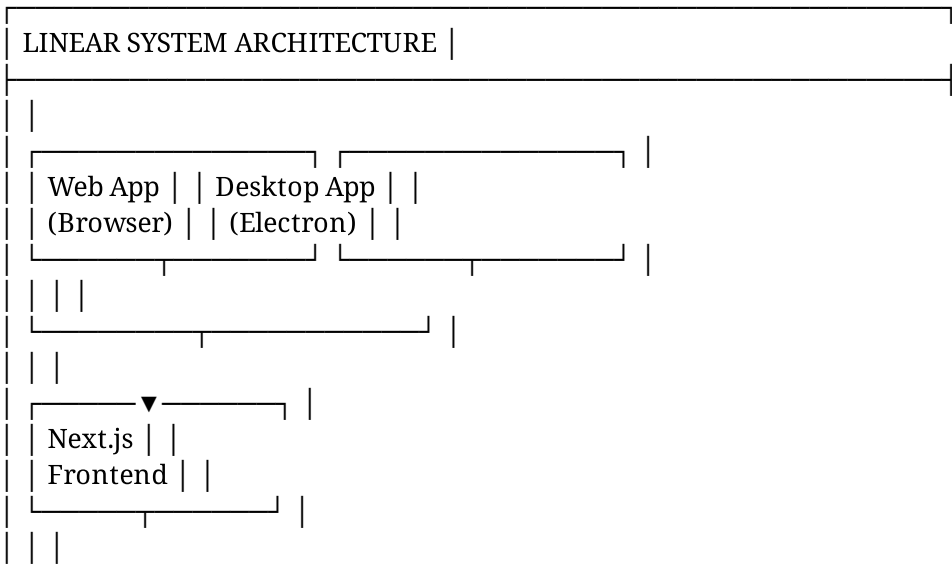
### Infraestrutura:

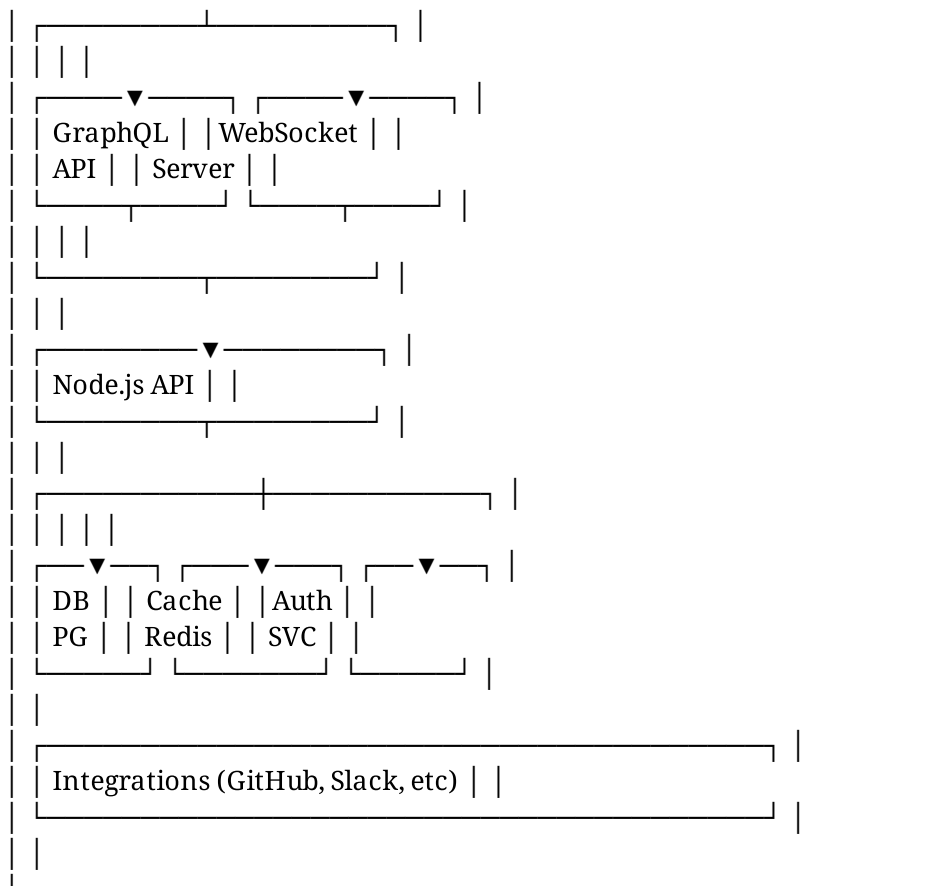
- AWS/GCP para hosting
- CloudFlare para CDN
- Auth0/Firebase para autenticação
- Stripe para billing

### Integrações:

- GitHub API (Octokit)
- Slack API
- Figma API
- Sentry SDK

## Componentes Principais





## □ Estrutura de Features Principais

### 1. Issue Management (Core)

#### Features:

- ✓ Criar issues via UI, CLI, Slack bot
- ✓ Atribuir prioridade (Urgent, High, Medium, Low)
- ✓ Estados customizáveis (Backlog, Todo, In Progress, In Review, Done, Cancelled)
- ✓ Ciclos (Sprints) com auto-rollover
- ✓ Labels e filtros avançados
- ✓ Full-text search com filtros
- ✓ Keyboard shortcuts (⌘K para abrir palette)

#### Requisitos Técnicos:

Issue Schema:

```

{
  id: string (LIN-123)
  title: string (max 255)
  description: string (markdown)
  priority: enum ['urgent', 'high', 'medium', 'low']
  status: string (customizable)
  assignee: User
  labels: Tag[]
  cycle: Cycle
}
  
```

```
project: Project
createdAt: timestamp
updatedAt: timestamp
comments: Comment[]
attachments: File[]
integrations: {
  github: { pr, branch, commit }
  figma: { link, lastChecked }
}
```

## 2. Project & Roadmap Management

### Features:

- ✓ Projects como containers para iniciativas
- ✓ Milestones para breaking down projects
- ✓ Roadmap visual com timeline
- ✓ Progreso automático baseado em issues
- ✓ Dependency tracking
- ✓ Lead time analytics

### Requisitos Técnicos:

Project Schema:

```
{
  id: string
  name: string
  description: string
  status: enum ['planned', 'active', 'paused', 'completed']
  progress: number (0-100, calculated)
  owner: User
  members: User[]
  milestones: Milestone[]
  issues: Issue[]
  startDate: date
  targetDate: date
  visibility: enum ['private', 'internal', 'public']
}
```

## 3. Ciclos/Sprints

### Features:

- ✓ Sprints configurable (1-4 semanas)
- ✓ Auto-rollover de issues incompletas
- ✓ Velocity tracking
- ✓ Burndown charts
- ✓ Planejamento de próximos ciclos

### Requisitos Técnicos:

Cycle Schema:

```
{
```

```
id: string
number: number
name: string
startDate: date
endDate: date
issues: Issue[]
completedCount: number (calculated)
totalCount: number (calculated)
status: enum ['planned', 'active', 'completed']
autoRollover: boolean
}
```

## 4. Colaboração Real-time

### Features:

- ✓ Comentários com @mentions
- ✓ Threading de comentários
- ✓ Reações em comments (emoji)
- ✓ Notificações via Slack/Email
- ✓ Atividades em tempo real (who viewed, who's typing)
- ✓ Histórico de mudanças completo

## 5. Integração GitHub/GitLab

### Features:

- ✓ Auto-linking de PRs para issues
- ✓ Status sync (PR opened → issue muda para "In Review")
- ✓ Auto-close quando PR é merged
- ✓ Commits linked a issues
- ✓ Branch naming conventions
- ✓ Merge request templates

### Fluxo de Integração:

User cria issue LIN-123

User cria PR com "linear: LIN-123" na descrição

Linear detecta e linka automaticamente

Quando PR é merged → LIN-123 muda para "Done"

GitHub mostra "Closes LIN-123" no description

Team vê contexto completo em ambas plataformas

## 6. Automações

### Features:

- ✓ Auto-assign baseado em regras
- ✓ Auto-status transitions
- ✓ Auto-archive de issues antigas
- ✓ Batch operations
- ✓ Custom workflows

### Exemplos de Automação:

Regra 1:

Trigger: Issue labeled "bug"

Action: Assign to @support-team

Action: Set priority to "high"

Regra 2:

Trigger: PR merged no GitHub

Action: Set status to "Done"

Action: Notify author no Slack

Regra 3:

Trigger: No activity por 14 dias

Action: Mark as "On Hold"

## 7. Search & Filtering

### Features:

- ✓ Full-text search em titles, descriptions, comments
- ✓ Filtros avançados com operadores (is:, has:, created:)
- ✓ Saved views personalizadas
- ✓ Filtros por labels, assignee, status, priority
- ✓ Busca com Cmd+K (command palette)

### Query Examples:

is:urgent assigned:me status:"in progress"

has:attachment created:2024-01-01..2024-01-31

project:Platform label:ui,ux priority:high

## 8. Temas Personalizados

### Features:

- ✓ Light theme (padrão)
- ✓ Dark theme
- ✓ High-contrast theme (acessibilidade)
- ✓ Temas customizados por organização
- ✓ Sistema de cores baseado em LCH (uniforme visualmente)

### Geração de Temas:

Usuário escolhe 3 variáveis:

1. Base Color (cor principal)
2. Accent Color (cor de destaque)
3. Contrast Level (0-100)

Sistema gera 50+ variáveis CSS automaticamente usando espaço LCH para uniformidade perceptual

---

# ▯ Design System & UI/UX

## Princípios de Design

### 1. Velocidade Acima de Tudo

- Zero latência aparente
- Keyboard-first interface
- Atalhos para ações comuns

### 2. Claridade Visual

- Hierarquia clara
- Sem clutter
- Espaçamento consistente

### 3. Densidade de Informação

- Máxima informação com mínimo de cliques
- Compact mode para poder users
- Expandable details

### 4. Acessibilidade

- WCAG 2.1 AA compliance mínimo
- High-contrast mode
- Keyboard navigation total
- Screen reader friendly

## Componentes Core

Button (Primary, Secondary, Outline, Danger)

- └ States: Default, Hover, Active, Disabled, Loading
- └ Sizes: SM, MD, LG
- └ Icons: Left, Right, Only

Input Fields

- └ Text, Email, Number, Password
- └ Textarea
- └ Select/Dropdown
- └ Checkbox
- └ Radio
- └ Toggle
- └ Date Picker

Card Component

- └ Header (with title, actions)
- └ Body (content area)
- └ Footer (with actions)

Modal/Dialog

- └ Alert Modal
- └ Form Modal
- └ Confirmation Modal
- └ Custom Modal

Tabs & Navigation

- └ Horizontal tabs
- └ Sidebar navigation



- └─ Breadcrumbs
- └─ Pagination

#### Data Display

- └─ Table (sortable, filterable)
- └─ List (virtualized para performance)
- └─ Timeline
- └─ Gantt Chart
- └─ Kanban Board

### Color System (LCH-based)

#### Light Theme:

- └─ Primary: #208080 (Teal)
- └─ Accent: #2180A1 (Blue-Teal)
- └─ Success: #32B8C6 (Teal Light)
- └─ Warning: #E68159 (Orange)
- └─ Error: #C01540 (Red)
- └─ Background: #FCFCF9 (Cream)
- └─ Surface: #FFFFFF (White)
- └─ Text: #134252 (Dark Blue)

#### Dark Theme:

- └─ Primary: #32B8C6 (Teal Light)
- └─ Accent: #2DB2C5 (Teal Lighter)
- └─ Success: #32B8C6
- └─ Warning: #E68159
- └─ Error: #FF5459
- └─ Background: #1F2121 (Very Dark)
- └─ Surface: #262828 (Dark)
- └─ Text: #F5F5F5 (Light Gray)

### Typography

#### Display (Headings): Inter Display 600-700

- └─ H1: 30px (display level)
- └─ H2: 24px (section headers)
- └─ H3: 20px (subsection)
- └─ H4: 18px (labels)

#### Body: Inter 400-500

- └─ Regular: 14px
- └─ Large: 16px
- └─ Small: 12px

#### Mono: Berkeley Mono

- └─ For code, issue IDs, timestamps
- └─ Base: 13px

## Layout & Spacing

Grid System: 4px base unit

- └─ Spacing: 4, 8, 12, 16, 20, 24, 32, 48px
- └─ Border Radius: 4, 6, 8, 12px
- └─ Gap/Margin: Consistent multiples of 4

Density Levels:

- └─ Compact: 4px gaps, 8px padding
- └─ Normal: 8px gaps, 12px padding
- └─ Spacious: 12px gaps, 16px padding

## Responsive Design

Breakpoints:

- └─ Mobile: 320px - 639px
- └─ Tablet: 640px - 1023px
- └─ Desktop: 1024px - 1279px
- └─ Wide: 1280px+

Layout Adjustments:

- └─ Mobile: Stack vertical, sidebar collapses
- └─ Tablet: Sidebar modal, 2-column layout
- └─ Desktop: Fixed sidebar, full 3-4 column

---

## □ Segurança & Autenticação

### Autenticação

- ✓ OAuth 2.0 (Google, GitHub, Microsoft)
- ✓ Email/Password com 2FA obrigatório
- ✓ SSO para empresas (SAML, OAuth)
- ✓ Sessions com JWT tokens
- ✓ Refresh tokens com rotação automática

### Autorização (RBAC)

Roles por Workspace:

- └─ Admin: Full access, manage team, billing
- └─ Member: Create/edit issues, full collaboration
- └─ Guest: Read-only, limited collaboration
- └─ Bot: API-only, specific permissions

Issue-Level Permissions:

- └─ Private (only assignee + author)
- └─ Team (team members)
- └─ Public (workspace)

## Dados & Privacidade

- ✓ Criptografia em trânsito (TLS 1.2+)
- ✓ Criptografia em repouso para dados sensíveis
- ✓ GDPR compliance
- ✓ Data export em JSON/CSV
- ✓ Soft delete com retenção 30 dias
- ✓ Audit logs para ações críticas

---

## Analytics & Metrics

### Métricas de Produto

#### Team Metrics:

- Velocity (issues/cycle)
- Cycle time (created → done)
- Lead time (prioritized → done)
- Burndown rate
- On-time completion rate

#### Queries:

GET /api/analytics/velocity?cycle\_id=CY-1&team\_id=TM-1

Response: {

totalIssues: 25,

completedIssues: 18,

velocity: 0.72,

trend: "↑ 12% vs ciclo anterior"

}

GET /api/analytics/cycle-time?from=2024-01-01&to=2024-01-31

Response: {

avgCycleTime: "4.2 days",

median: "3.5 days",

p95: "8.2 days"

}

### Instrumentation

- ✓ Sentry para error tracking
- ✓ PostHog/Mixpanel para product analytics
- ✓ Custom metrics (Prometheus)
- ✓ Performance monitoring (Core Web Vitals)

---

## Integrações

## GitHub

### Funcionalidades:

- Auto-sync PRs ↔ Issues
- Branch naming conventions
- Commit message parsing
- Deploy tracking

### Setup:

1. Usuário clica "Connect GitHub"
2. OAuth flow em [github.com](https://github.com)
3. Linear app request permissions
4. Usa GitHub App (não OAuth token pessoal)
5. Webhook para eventos (push, PR, release)

## Slack

### Funcionalidades:

- Notificações de updates
- Criar issues via Slack bot
- Abrir linear.app/issue direto do Slack
- /linear shortcut

### Exemplo:

/linear create "Fix login button on mobile"

- Bot cria issue
- Thread mostra LIN-123
- User clica para abrir no Linear

Quando issue é movida para "Done":

- Bot notifica no Slack
- Mostra burndown do ciclo

## Figma

### Funcionalidades:

- Embed design previews em issues
- Link designs em issues
- Notificações de mudanças
- Design comments sync

## Sentry

### Funcionalidades:

- Auto-create issues de erros críticos
  - Stack trace no Linear
  - Link issue → sentry issue
  - Performance metrics
-

# □ Plano de Implementação

## Phase 1: MVP (Weeks 1-8)

- ✓ Core issue management
- ✓ Basic UI (list view)
- ✓ Team management
- ✓ Autenticação

### Deliverables:

- Working web app
- Issue CRUD
- Basic filters
- Slack integration (notifications)

## Phase 2: Enhancement (Weeks 9-16)

- ✓ Board view (Kanban)
- ✓ Cycles (sprints)
- ✓ Projects
- ✓ GitHub integration
- ✓ Automations

## Phase 3: Polish (Weeks 17-20)

- ✓ Performance optimization
- ✓ Accessibility audit
- ✓ Design refinement
- ✓ Desktop app (Electron)
- ✓ Launch preps

## Phase 4: Post-Launch (Weeks 21+)

- ✓ Beta feedback
- ✓ Bug fixes
- ✓ V1.1 features
- ✓ Enterprise features (SSO, etc)

---

# □ Success Metrics

## Adoption Metrics

- Signups por semana: Target 500+ (beta)
- Active teams: Target 100+ (beta)
- Issues created/day: Target 10,000+
- DAU/MAU ratio: Target 70%+

## Engagement Metrics

- Avg issues/user/month: Target 20+
- Avg team size: Target 8 pessoas
- Retention day 7: Target 60%+
- Retention day 30: Target 40%+

## Business Metrics

- Monthly Recurring Revenue (MRR)
- Net Revenue Retention (NRR)
- Customer Acquisition Cost (CAC)
- Customer Lifetime Value (LTV)

---

## ▯ Requisitos Técnicos Não-Funcionais

### Performance

Requisitos:

- └─ Página load: < 2s (Lighthouse 90+)
- └─ API response: < 100ms (p95)
- └─ Search: < 200ms (full-text)
- └─ WebSocket latency: < 50ms
- └─ Database query: < 50ms (p95)

Optimization:

- └─ Code splitting
- └─ Lazy loading de features
- └─ Image optimization
- └─ Database indexing
- └─ Redis caching layer
- └─ CDN para assets

### Escalabilidade

Targets:

- └─ 10,000+ DAU
- └─ 1,000,000+ issues total
- └─ 99.9% uptime SLA
- └─ Auto-scaling horizontal

Architecture:

- └─ Stateless API servers
- └─ Load balancing
- └─ Database replication
- └─ Cache invalidation strategy

## Confiabilidade

### Requisitos:

- └ Error rate: < 0.1%
- └ Data durability: RPO 1 hour
- └ Disaster recovery: RTO 4 hours
- └ Backup: Daily, 30 dias retention
- └ Monitoring: 24/7 alerts

### Testing:

- └ Unit test coverage: 70%+
  - └ Integration tests: Critical paths
  - └ Load testing: 10,000 concurrent
  - └ Chaos engineering: Resilience
- 

## ▯ Especificações de API

### REST Endpoints (Principal API)

## Issues

GET /api/issues # List com filters

POST /api/issues # Create

GET /api/issues/:id # Get

PATCH /api/issues/:id # Update

DELETE /api/issues/:id # Archive

## Comments

POST /api/issues/:id/comments # Add comment

PATCH /api/comments/:id # Edit

DELETE /api/comments/:id # Delete

## Projects

GET /api/projects # List

POST /api/projects # Create

PATCH /api/projects/:id # Update

GET /api/projects/:id/issues # Issues in project

## Cycles

GET /api/cycles # List

POST /api/cycles # Create

PATCH /api/cycles/:id # Update

POST /api/cycles/:id/complete # Complete cycle

# Teams

GET /api/teams # List

GET /api/teams/:id/members # Team members

POST /api/teams/:id/members # Invite

# Analytics

GET /api/analytics/velocity # Velocity

GET /api/analytics/cycle-time # Cycle time

GET /api/analytics/burndown # Burndown chart

## GraphQL Schema (Alternativa)

```
type Issue {
  id: ID!
  identifier: String! # LIN-123
  title: String!
  description: String
  priority: Priority!
  status: Status!
  assignee: User
  labels: [Label!]
  project: Project
  cycle: Cycle
  createdAt: DateTime!
  updatedAt: DateTime!
  comments: [Comment!]
  linkedIssues: [Issue!]
}

type Query {
  issues(filter: IssueFilter, sort: IssueSort): [Issue!]!
  issue(id: ID!): Issue
  projects: [Project!]!
  cycles: [Cycle!]!
  analytics(from: DateTime!, to: DateTime!): Analytics!
}

type Mutation {
  createIssue(input: CreateIssueInput!): Issue!
  updateIssue(id: ID!, input: UpdateIssueInput!): Issue!
  addComment(issueId: ID!, content: String!): Comment!
}
```

---



## ▮ Roadmap Futuro (v1.1+)

### Q2 2026

- ☐ Mobile app (React Native)
- ☐ Collaborative editing para descriptions
- ☐ Advanced reporting
- ☐ Jira migration tools

### Q3 2026

- ☐ Webhooks customizados
- ☐ API public
- ☐ Custom fields
- ☐ Time tracking integration

### Q4 2026

- ☐ AI-powered issue classification
- ☐ Natural language search
- ☐ Smart automation suggestions
- ☐ Enterprise features (audit logs, SSO)

---

## ▮ Glossário

- **Cycle:** Sprint ou período de trabalho (1-4 semanas)
- **Issue:** Unidade de trabalho (bug, feature, task)
- **Project:** Container para iniciativas relacionadas
- **Milestone:** Breaking point no project timeline
- **Lead Time:** Tempo de criação até conclusão
- **Velocity:** Issues completadas por cycle
- **Burndown:** Progresso visual do cycle

---

## ✓ Checklist de Implementação

### Backend

- ☐ Database schema completo
- ☐ Authentication flow (OAuth + 2FA)
- ☐ Issue CRUD endpoints
- ☐ Real-time WebSocket server
- ☐ Integração GitHub API
- ☐ Integração Slack API
- ☐ Job queue (automações)
- ☐ Caching layer (Redis)
- ☐ Error handling e logging
- ☐ Tests (70%+ coverage)

## Frontend

- ☐ Design system implementado
- ☐ List view (virtualized)
- ☐ Board view (Kanban)
- ☐ Timeline view
- ☐ Search/filter advanced
- ☐ Command palette (Cmd+K)
- ☐ Real-time updates (WebSocket)
- ☐ Offline support
- ☐ Keyboard shortcuts
- ☐ Accessibility audit

## DevOps

- ☐ CI/CD pipeline
- ☐ Staging environment
- ☐ Production monitoring
- ☐ Error tracking (Sentry)
- ☐ Analytics setup
- ☐ Backup strategy
- ☐ Disaster recovery plan

## Marketing & Launch

- ☐ Landing page
- ☐ Onboarding flow
- ☐ Demo video
- ☐ Documentation
- ☐ Community setup
- ☐ Beta user outreach
- ☐ Press release
- ☐ Twitter/LinkedIn campaign

---

### Próximas Etapas:

1. Revisão com stakeholders
2. Design system final (Figma)
3. Kickoff development (Sprint Planning)
4. Setup CI/CD e monitoramento