Business Requirements Document: Task Management System

1. Project Overview

Project Name: TaskFlow - Collaborative Task Management System

Objective: Develop a comprehensive web-based task management platform for teams

Business Context: Enable efficient project management and team collaboration for modern workplaces

2. Stakeholders

Business Team:

- Project Managers
- Team Leaders
- Department Heads

Technical Team:

- Full-Stack Developers
- DevOps Engineers
- UX/UI Designers

End Users:

- Team Members
- Project Managers
- Administrators

3. Functional Requirements

3.1 User Management

- User registration and authentication
- Role-based access control (Admin, Manager, Member)
- User profiles with avatar, contact information, and preferences
- Team creation and member management

3.2 Project Management

- Create, update, and delete projects
- Project attributes: name, description, start/end dates, status, priority
- Project templates for common workflows
- Project archiving and restoration

3.3 Task Management

- Create, assign, and track tasks within projects
- Task attributes: title, description, assignee, due date, priority, status, labels
- Task dependencies and subtasks
- Task comments and file attachments
- Time tracking and logging

3.4 Board Views

- Kanban board view with customizable columns
- List view with sorting and filtering options
- Calendar view for deadline tracking
- Gantt chart for project timeline visualization

3.5 Collaboration Features

- Real-time notifications
- Task comments and mentions
- File sharing and document collaboration
- Activity feeds for projects and tasks

4. Non-Functional Requirements

- Support for 1,000+ concurrent users
- 99.9% uptime availability
- Response time under 1 second for standard operations
- Mobile-responsive design
- Offline capability for task viewing
- GDPR compliance for data protection

5. Technical Specifications

5.1 Frontend

- React 18 with TypeScript
- Material-UI (MUI) component library
- Redux Toolkit for state management
- React Query for server state management
- React Router for navigation

5.2 Backend

- Node.js with Express.js framework
- GraphQL API with Apollo Server
- PostgreSQL database with Prisma ORM
- Redis for session management and caching
- Socket.io for real-time updates

5.3 Infrastructure

- Docker containerization
- AWS/Azure cloud deployment
- CI/CD pipeline with GitHub Actions
- Monitoring with Application Insights

6. Data Model

Users:

- id: UUID
- email: string (unique)
- username: string (unique)
- full_name: string
- avatar_url: string
- role: enum(admin, manager, member)
- created_at: timestamp
- updated_at: timestamp

Projects:

- id: UUID

- name: string
- description: text
- owner_id: UUID (foreign key to Users)
- start_date: date
- end_date: date
- status: enum(planning, active, completed, archived)
- priority: enum(low, medium, high, urgent)
- created_at: timestamp
- updated_at: timestamp

Tasks:

- id: UUID
- title: string
- description: text
- project_id: UUID (foreign key to Projects)
- assignee_id: UUID (foreign key to Users)
- reporter_id: UUID (foreign key to Users)
- due_date: datetime
- priority: enum(low, medium, high, urgent)
- status: enum(todo, in_progress, review, done)
- estimated_hours: decimal
- actual_hours: decimal
- parent_task_id: UUID (self-referencing for subtasks)
- created_at: timestamp
- updated_at: timestamp

TaskComments:

- id: UUID
- task_id: UUID (foreign key to Tasks)
- author_id: UUID (foreign key to Users)
- content: text
- created_at: timestamp
- updated_at: timestamp

7. API Endpoints

7.1 Authentication

- POST /auth/login - User authentication
- POST /auth/register - User registration
- POST /auth/logout - User logout
- POST /auth/refresh - Token refresh

7.2 Projects

- GET /api/projects - List user projects
- POST /api/projects - Create new project
- GET /api/projects/:id - Get project details
- PUT /api/projects/:id - Update project
- DELETE /api/projects/:id - Delete project

7.3 Tasks

- GET /api/projects/:projectId/tasks - List project tasks
- POST /api/projects/:projectId/tasks - Create new task
- GET /api/tasks/:id - Get task details
- PUT /api/tasks/:id - Update task
- DELETE /api/tasks/:id - Delete task
- POST /api/tasks/:id/comments - Add task comment

7.4 Users

- GET /api/users/profile - Get current user profile
- PUT /api/users/profile - Update user profile
- GET /api/projects/:projectId/members - Get project members

8. Security Requirements

- JWT-based authentication
- RBAC (Role-Based Access Control)
- Input validation and sanitization
- SQL injection protection
- XSS protection
- HTTPS enforcement
- Rate limiting (100 requests per minute per user)
- Password encryption using bcrypt

9. Integration Requirements

- Email notifications (SendGrid/AWS SES)
- File storage (AWS S3/Azure Blob Storage)
- Calendar integration (Google Calendar, Outlook)
- Slack/Teams integration for notifications
- Time tracking tools integration

10. Performance Requirements

- Page load time: < 2 seconds
- API response time: < 500ms for CRUD operations
- Database query optimization
- Caching strategy for frequently accessed data
- CDN for static assets

11. Testing Requirements

- Unit tests (Jest/Vitest) - minimum 80% coverage
- Integration tests for API endpoints
- End-to-end tests (Cypress/Playwright)
- Performance testing
- Security testing

12. Timeline and Milestones

Phase 1 - Core Foundation (6 weeks):

- User authentication and management
- Basic project and task CRUD operations
- Simple list and board views

Phase 2 - Advanced Features (4 weeks):

- Real-time collaboration
- File attachments
- Advanced filtering and search
- Time tracking

Phase 3 - Integration & Polish (4 weeks):

- Third-party integrations
- Mobile optimization
- Performance optimization
- Security hardening

Phase 4 - Deployment & Monitoring (2 weeks):

- Production deployment
- Monitoring and analytics setup
- User training and documentation

13. Success Criteria

- System handles 1,000+ concurrent users without performance degradation
- 95% user satisfaction score
- Task creation time reduced by 50% compared to current manual processes
- 99.9% uptime achievement
- Zero critical security vulnerabilities