**TASK MANAGEMENT SYSTEM - PROJECT WORKFLOW**

**📋 PROJECT OVERVIEW**

**Project:** Task Management System  
**Team Size:** 8 Members  
**Duration:** October - December 2025  
**Technology Stack:** Spring Boot, React.js, MySQL, JWT Authentication

**👥 TEAM ROLE SUMMARY**

| Member | Role | Key Responsibilities |
| --- | --- | --- |
| **Member 1** | Database & Entity Designer | Database schema, JPA entities, repositories |
| **Member 2** | Task Management & Business Logic | Task CRUD operations, filtering, pagination |
| **Member 3** | Authentication & Security | JWT authentication, Spring Security, CORS |
| **Member 4** | Special Features & Reports | Email, scheduler, PDF/CSV export, admin APIs |
| **Member 5** | React Setup & Routing | App structure, routing, context, Axios setup |
| **Member 6** | Authentication Pages | Login/Register UI, form validation, auth integration |
| **Member 7** | Dashboard & Task Management | Task CRUD UI, dashboard, search/filter UI |
| **Member 8** | Admin Dashboard & UI | Charts, responsive design, styling, admin features |

**🔄 COMPLETE WORKFLOW BREAKDOWN**

**PHASE 1: FOUNDATION SETUP (Week 1-2)**

**🎯 STARTUP SEQUENCE - WEEK 1**

**IMMEDIATE STARTERS (Day 1-3):**

**👤 Member 1 - Database & Entity Designer**

* Create GitHub repository
* Design ER Diagram (User ↔ Task relationships)
* Write database schema.sql
* Create MySQL database taskdb
* Implement core entities: User.java, Task.java, Role.java
* Create repositories: UserRepository.java, TaskRepository.java

**👤 Member 5 - React Setup & Routing**

* Initialize React app: npx create-react-app taskmanager-frontend
* Install dependencies: react-router-dom, axios, bootstrap
* Setup basic routing structure in App.js
* Create AuthContext.js for global state management
* Implement ProtectedRoute.js components

**DELIVERABLES BY END OF WEEK 1:**

* ✅ GitHub repository with initial structure
* ✅ Database schema and entities
* ✅ React app with routing foundation
* ✅ Basic project documentation

**🎯 CORE DEVELOPMENT - WEEK 2**

**PARALLEL STREAMS ACTIVATED:**

**BACKEND STREAM:**  
**👤 Member 2 - Task Management** (Starts when entities are ready)

* Create TaskService.java with CRUD methods
* Implement filtering and sorting logic
* Build TaskController.java with REST endpoints
* Create DTOs: TaskRequest.java, TaskResponse.java
* Test APIs with Postman

**👤 Member 3 - Authentication** (Starts when entities are ready)

* Configure SecurityConfig.java with Spring Security
* Implement JwtAuthenticationFilter.java
* Create AuthController.java with login/register endpoints
* Setup JwtUtil.java for token management
* Configure CORS for React frontend

**FRONTEND STREAM:**  
**👤 Member 6 - Authentication Pages** (Starts when routing is ready)

* Build Login.js with form validation
* Build Register.js with input validation
* Create form validation utilities in validators.js
* Connect to backend auth endpoints using Axios
* Implement loading states and error handling

**DELIVERABLES BY END OF WEEK 2:**

* ✅ Backend CRUD APIs functional
* ✅ JWT authentication working
* ✅ Login/Register UI complete
* ✅ Frontend-backend connection established

**PHASE 2: CORE FEATURE DEVELOPMENT (Week 3-4)**

**🎯 INTEGRATION & FEATURES - WEEK 3**

**ALL MEMBERS ACTIVE:**

**👤 Member 7 - Task Management UI** (Starts when Task APIs are ready)

* Create Dashboard.js with task statistics
* Build TaskList.js for displaying tasks
* Implement TaskForm.js for add/edit operations
* Create TaskFilters.js for search and filtering
* Develop useTasks.js custom hook

**👤 Member 4 - Special Features** (Starts when core backend is ready)

* Implement EmailService.java with Spring Boot Mail
* Create TaskScheduler.java for overdue task checking
* Build ReportService.java for PDF/CSV generation
* Develop AdminController.java for statistics
* Setup GlobalExceptionHandler.java

**👤 Member 8 - Admin Dashboard** (Starts when admin APIs are ready)

* Create AdminDashboard.js with charts
* Implement responsive design with Bootstrap
* Build chart components using Recharts/Chart.js
* Create reusable UI components
* Apply overall styling and themes

**INTEGRATION FOCUS:**

* Member 6 ↔ Member 3: Auth flow testing
* Member 7 ↔ Member 2: Task CRUD integration
* Member 8 ↔ Member 4: Admin features integration

**DELIVERABLES BY END OF WEEK 3:**

* ✅ Full task management UI functional
* ✅ Email notifications and scheduler working
* ✅ Admin dashboard with basic charts
* ✅ End-to-end user workflows tested

**🎯 ENHANCEMENT & POLISH - WEEK 4**

**FINAL INTEGRATION AND REFINEMENT:**

**CROSS-TEAM COLLABORATION:**

* **API Contract Verification:** All members verify endpoint compatibility
* **Error Handling:** Consistent error responses frontend/backend
* **User Experience:** Loading states, success messages, error displays
* **Responsive Design:** Mobile-friendly interface testing

**SPECIAL FEATURES COMPLETION:**

* Member 4: PDF/CSV export functionality
* Member 8: Advanced charts and analytics
* Member 7: Enhanced search and pagination
* Member 3: Role-based access control refinement

**DELIVERABLES BY END OF WEEK 4:**

* ✅ All 5 special features implemented
* ✅ Responsive design completed
* ✅ Comprehensive error handling
* ✅ Full application testing

**PHASE 3: TESTING & DEPLOYMENT (Week 5-6)**

**🎯 FINAL POLISH - WEEK 5-6**

**COMPREHENSIVE TESTING:**

text

TESTING AREAS:

✓ Unit Testing: Service layer methods

✓ Integration Testing: API endpoints

✓ UI Testing: User interactions and workflows

✓ Security Testing: Authentication and authorization

✓ Performance Testing: Database queries and page loads

✓ Cross-browser Testing: Compatibility verification

**DEPLOYMENT PREPARATION:**

* Environment configuration (dev/prod)
* Database production setup
* Build optimization
* Documentation completion

**FINAL DELIVERABLES:**

* ✅ Fully functional web application
* ✅ Complete documentation
* ✅ Deployment-ready code
* ✅ Project presentation materials

**🔗 CRITICAL DEPENDENCY MAP**

**TECHNICAL DEPENDENCIES**

text

Member 1 (Database)

↓

Member 2 (Task APIs) → Member 7 (Task UI)

Member 3 (Auth APIs) → Member 6 (Auth UI)

↓

Member 4 (Admin APIs) → Member 8 (Admin UI)

↓

Member 5 (React Setup) → All Frontend Members

**INTEGRATION CHECKPOINTS**

**Checkpoint 1 (Week 2):** Backend Entities → Frontend Routing  
**Checkpoint 2 (Week 3):** Auth APIs → Auth UI | Task APIs → Task UI  
**Checkpoint 3 (Week 4):** Admin APIs → Admin UI | All Features Integration  
**Checkpoint 4 (Week 5):** End-to-End Testing | Bug Fixing  
**Checkpoint 5 (Week 6):** Final Polish | Documentation

**⚠️ RISK MITIGATION STRATEGIES**

**POTENTIAL BLOCKERS & SOLUTIONS**

| Risk | Impact | Mitigation Strategy |
| --- | --- | --- |
| **Database Schema Changes** | High | Member 1 freezes schema by Week 2; communicate changes early |
| **API Specification Changes** | High | Maintain API documentation; version endpoints |
| **Integration Issues** | Medium | Weekly integration testing; mock APIs during development |
| **Feature Scope Creep** | Medium | Stick to defined feature list; postpone enhancements |
| **Technical Knowledge Gaps** | Low | Pair programming; knowledge sharing sessions |

**COMMUNICATION PROTOCOLS**

* **Daily Standups:** Quick progress updates
* **Weekly Sync Meetings:** Cross-team integration check
* **GitHub Issues:** Track tasks and bugs
* **API Documentation:** Maintain updated Postman collection

**✅ SUCCESS CRITERIA**

**INDIVIDUAL SUCCESS METRICS**

* All assigned files completed and functional
* Features work in integrated environment
* Code follows team standards and best practices
* Active participation in team collaboration
* Tasks completed within estimated timelines

**PROJECT SUCCESS METRICS**

* All 5 special features implemented and working
* Full CRUD operations functional end-to-end
* Responsive and user-friendly interface
* Secure authentication and authorization
* Comprehensive documentation
* Successful demo and presentation