


División de Trabajo - API Gestión de Proyectos

 **Samuel Gomez - Módulo de Autenticación y Usuarios**

Responsabilidades

- Configuración inicial del proyecto
- Sistema de autenticación completo
- Gestión de usuarios y roles
- Middleware de seguridad

Tareas Específicas

1. Configuración Base

- ☐ Configurar servidor Express
- ☐ Conexión a MongoDB
- ☐ Configuración de middlewares globales
- ☐ Manejo de errores centralizado
- ☐ Rate limiting

2. Modelos

- ☐ `User.model.js` - Modelo de usuarios
- ☐ `Role.model.js` - Modelo de roles

3. Autenticación

- ☐ `auth.controller.js` - Controlador de autenticación
- ☐ `auth.routes.js` - Rutas de autenticación
- ☐ `auth.service.js` - Servicios de autenticación
- ☐ `auth.validator.js` - Validaciones de autenticación

4. Gestión de Usuarios

- ☐ `user.controller.js` - Controlador de usuarios
- ☐ `user.routes.js` - Rutas de usuarios
- ☐ `user.validator.js` - Validaciones de usuarios

5. Roles y Permisos

- [] role.controller.js - Controlador de roles
- [] role.routes.js - Rutas de roles
- [] role.middleware.js - Middleware de autorización

6. Servicios

- [] email.service.js - Servicio de emails
- [] encryption.util.js - Utilidades de encriptación

Endpoints a desarrollar

```
POST /api/auth/register
POST /api/auth/login
POST /api/auth/refresh
POST /api/auth/logout
POST /api/auth/forgot-password
POST /api/auth/reset-password
POST /api/auth/verify-email # NUEVA RUTA
POST /api/auth/resend-verification # NUEVA RUTA
```

```
GET /api/users
GET /api/users/profile
PUT /api/users/profile
DELETE /api/users/:id
PUT /api/users/:id/role
```

```
GET /api/roles
POST /api/roles
PUT /api/roles/:id
DELETE /api/roles/:id
```



Franklin Peña - Módulo de Gestión Base y Estados



Responsabilidades

- Sistema de categorías
- Gestión de estados
- Funcionalidades de upload
- Utilidades generales



Tareas Específicas

1. Modelos

- [] Category.model.js - Modelo de categorías
- [] State.model.js - Modelo de estados

2. Categorías

- [] `category.controller.js` - Controlador de categorías
- [] `category.routes.js` - Rutas de categorías

3. Estados

- [] `state.controller.js` - Controlador de estados
- [] `state.routes.js` - Rutas de estados

4. Sistema de Upload

- [] `upload.controller.js` - Controlador de uploads
- [] `upload.routes.js` - Rutas de uploads
- [] `upload.service.js` - Servicio de uploads
- [] `upload.middleware.js` - Middleware de multer
- [] Configuración de multer

5. Utilidades

- [] `response.util.js` - Utilidades de respuesta
- [] `validation.util.js` - Utilidades de validación
- [] `constants.js` - Constantes del sistema

6. Middlewares

- [] `validation.middleware.js` - Middleware de validación
- [] `error.middleware.js` - Middleware de errores

Endpoints a desarrollar

```
GET /api/categories
POST /api/categories
PUT /api/categories/:id
DELETE /api/categories/:id
```

```
GET /api/states/projects
GET /api/states/tasks
POST /api/states
PUT /api/states/:id
DELETE /api/states/:id
```

```
POST /api/upload/avatar # NUEVA RUTA
POST /api/upload/document # NUEVA RUTA
DELETE /api/upload/:filename # NUEVA RUTA
```



Mariana Gomez - Módulo de Proyectos

Responsabilidades

- **Gestión completa de proyectos**
- **Sistema de miembros**
- **Relaciones proyecto-usuario**

Tareas Específicas

1. Modelos

- ☐ `Project.model.js` - Modelo de proyectos (completo con todas las relaciones)

2. Proyectos

- ☐ `project.controller.js` - Controlador de proyectos
- ☐ `project.routes.js` - Rutas de proyectos
- ☐ `project.validator.js` - Validaciones de proyectos

3. Funcionalidades Específicas

- ☐ Gestión de miembros del proyecto
- ☐ Control de permisos por proyecto
- ☐ Cambio de estados de proyecto
- ☐ Filtros y búsquedas avanzadas
- ☐ Reportes de progreso

4. Validaciones Complejas

- ☐ Validación de permisos de proyecto
- ☐ Validación de transiciones de estado
- ☐ Validación de miembros y roles

Endpoints a desarrollar

```
GET /api/projects
POST /api/projects
GET /api/projects/:id
PUT /api/projects/:id
DELETE /api/projects/:id
GET /api/projects/:id/details
POST /api/projects/:id/members
DELETE /api/projects/:id/members/:userId
PUT /api/projects/:id/members/:userId/role
PUT /api/projects/:id/status
GET /api/projects/:id/progress
GET /api/projects/my-projects
GET /api/projects/statistics # NUEVA RUTA
```



Mariam Pizza - Módulo de Tareas, Comentarios e IA



Responsabilidades

- **Gestión de tareas**
- **Sistema de comentarios**
- **Integración con IA**
- **Funcionalidades avanzadas**



Tareas Específicas

1. Modelos

- `[] Task.model.js` - Modelo de tareas
- `[] Comment.model.js` - Modelo de comentarios

2. Tareas

- `[] task.controller.js` - Controlador de tareas
- `[] task.routes.js` - Rutas de tareas
- `[] task.validator.js` - Validaciones de tareas

3. Comentarios

- `[] comment.controller.js` - Controlador de comentarios
- `[] comment.routes.js` - Rutas de comentarios

4. Inteligencia Artificial

- `[] ai.controller.js` - Controlador de IA
- `[] ai.routes.js` - Rutas de IA
- `[] ai.service.js` - Servicio de IA (integración OpenAI)

5. Funcionalidades Avanzadas

- `[]` Sistema de dependencias entre tareas
- `[]` Notificaciones automáticas
- `[]` Análisis de progreso con IA



Endpoints a desarrollar

```
GET  /api/projects/:projectId/tasks
POST /api/projects/:projectId/tasks
GET  /api/tasks/:id
PUT  /api/tasks/:id
```

```
DELETE /api/tasks/:id
PUT /api/tasks/:id/status
PUT /api/tasks/:id/assign
GET /api/tasks/my-tasks
GET /api/tasks/:id/dependencies

GET /api/projects/:id/comments
POST /api/projects/:id/comments
PUT /api/comments/:id
DELETE /api/comments/:id

POST /api/ai/generate-tasks
POST /api/ai/analyze-project
POST /api/ai/estimate-time
POST /api/ai/generate-summary
POST /api/ai/suggest-improvements
```

Coordinación y Dependencias

Orden de Desarrollo Recomendado:

1. **Samuel** → Base del proyecto (autenticación, usuarios, roles)
2. **Franklin** → Categorías, estados y uploads (depende de usuarios)
3. **Mariana** → Proyectos (depende de usuarios, categorías, estados)
4. **Mariam** → Tareas y comentarios (depende de proyectos), IA (independiente)

Reuniones de Sincronización:

- **Daily:** 15 min cada mañana
- **Review:** Cada viernes para revisar progreso
- **Integration:** Cuando se complete cada módulo

Herramientas de Colaboración:

- **Git:** Ramas por desarrollador (feature/samuel-auth, etc.)
- **Postman:** Colección compartida de endpoints
- **Slack/Discord:** Comunicación diaria
- **Trello/Jira:** Seguimiento de tareas

Estimación de Tiempo:

- **Cada módulo:** 2-3 semanas
- **Integración:** 1 semana
- **Testing:** 1 semana
- **Total del proyecto:** 8-10 semanas

