Development Setup Guide

Quick Start

Prerequisites Installation

1. Install PHP 8.1+

```bash
# macOS (using Homebrew)
brew install php@8.1

# Ubuntu/Debian
sudo apt update
sudo apt install php8.1 php8.1-cli php8.1-common php8.1-mysql php8.1-zip php8.1-gd php8.1-mb

# Windows
# Download from https://windows.php.net/download/
```

2. Install Composer

```bash
# macOS/Linux
curl -sS https://getcomposer.org/installer | php
sudo mv composer.phar /usr/local/bin/composer

# Windows
# Download from https://getcomposer.org/download/
```

3. Install Node.js 18+

```bash
# macOS (using Homebrew)
brew install node

# Ubuntu/Debian
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt-get install -y nodejs

# Windows
# Download from https://nodejs.org/
```

4. Install MySQL/PostgreSQL

```bash
# MySQL (macOS)
brew install mysql
brew services start mysql

# PostgreSQL (macOS)
brew install postgresql
brew services start postgresql
```

```
# Ubuntu/Debian
sudo apt install mysql-server
# or
sudo apt install postgresql postgresql-contrib
```

Project Setup

1. Clone and Install Dependencies

```
# Clone the repository
git clone <your-repo-url>
cd innotrack-web

# Install PHP dependencies
composer install

# Install Node.js dependencies
npm install
```

2. Environment Configuration

```
# Copy environment file
cp .env.example .env

# Generate application key
php artisan key:generate

# Configure database in .env file
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=innotrack_web
DB_USERNAME=root
DB_PASSWORD=your_password
```

3. Database Setup

```
# Create database
mysql -u root -p -e "CREATE DATABASE innotrack_web;"
# or for PostgreSQL
createdb innotrack_web

# Run migrations
php artisan migrate
```

```
# Seed database with sample data
php artisan db:seed
```

4. Start Development Servers

```
# Terminal 1: Start Laravel development server
php artisan serve

# Terminal 2: Start Vite development server
npm run dev
```

Development Workflow

Git Workflow

```
# Create feature branch
git checkout -b feature/your-feature-name

# Make changes and commit
git add .
git commit -m "feat: add your feature description"

# Push to remote
git push origin feature/your-feature-name

# Create pull request on GitHub/GitLab
```

Code Style

```
# Format PHP code
./vendor/bin/pint

# Format JavaScript/React code
npm run format

# Lint JavaScript/React code
npm run lint
```

Testing

```
# Run all tests
php artisan test

# Run tests with coverage
php artisan test --coverage
```

```
# Run specific test file
php artisan test tests/Feature/YourTest.php
```

Project Structure Overview

```
innotrack-web/
  app/
     Http/
        Controllers/
           Api/           # API Controllers (Member 1)
           Admin/         # Admin Controllers
           Dashboard/     # Dashboard Controllers
        Requests/          # Form Requests (Member 1)
        Resources/         # API Resources (Member 1)
     Models/                # Eloquent Models (COMPLETED  )
     Services/              # Business Logic
     Policies/              # Authorization Policies (Member 1)
  database/
     migrations/            # Database Migrations (COMPLETED  )
     seeders/               # Database Seeders
     factories/             # Model Factories
  resources/
     js/
         Components/        # Reusable React Components (Member 2)
         Pages/             # Page Components (Members 2,3,4,5)
         Layouts/           # Layout Components (Member 2)
         Hooks/             # Custom React Hooks
         Services/          # API Services
         Utils/             # Utility Functions
  tests/
      Feature/               # Feature Tests (All Members)
      Unit/                  # Unit Tests (All Members)
      Browser/               # Browser Tests
```

Key Development Areas by Member

Member 1: Backend Lead

Focus Areas:

- `app/Http/Controllers/Api/` - API Controllers
- `app/Http/Requests/` - Form Validation
- `app/Http/Resources/` - API Resources
- `app/Policies/` - Authorization
- `app/Services/` - Business Logic

Key Files to Create:

- `app/Http/Controllers/Api/ProgramController.php`
- `app/Http/Controllers/Api/FacilityController.php`
- `app/Http/Controllers/Api/ProjectController.php`
- `app/Http/Controllers/Api/ParticipantController.php`
- `app/Http/Controllers/Api/OutcomeController.php`

Member 2: Frontend Lead

Focus Areas:

- `resources/js/Components/` - Reusable Components
- `resources/js/Layouts/` - Layout Components
- `resources/js/Pages/` - Page Components
- `resources/js/Hooks/` - Custom Hooks

Key Files to Create:

- `resources/js/Components/DataTable.jsx`
- `resources/js/Components/Form.jsx`
- `resources/js/Components/Modal.jsx`
- `resources/js/Layouts/AdminLayout.jsx`
- `resources/js/Pages/Dashboard.jsx`

Member 3: Projects & Participants

Focus Areas:

- `resources/js/Pages/Projects/` - Project Pages
- `resources/js/Pages/Participants/` - Participant Pages
- `app/Http/Controllers/Api/ProjectController.php` (with Member 1)
- `app/Http/Controllers/Api/ParticipantController.php` (with Member 1)

Key Files to Create:

- `resources/js/Pages/Projects/Index.jsx`
- `resources/js/Pages/Projects/Create.jsx`
- `resources/js/Pages/Projects/Edit.jsx`
- `resources/js/Pages/Participants/Index.jsx`

Member 4: Facilities & Equipment

Focus Areas:

- `resources/js/Pages/Facilities/` - Facility Pages
- `resources/js/Pages/Equipment/` - Equipment Pages
- `app/Http/Controllers/Api/FacilityController.php` (with Member 1)
- `app/Http/Controllers/Api/EquipmentController.php` (with Member 1)

Key Files to Create:

- `resources/js/Pages/Facilities/Index.jsx`
- `resources/js/Pages/Facilities/Create.jsx`
- `resources/js/Pages/Equipment/Index.jsx`
- `resources/js/Pages/Equipment/Create.jsx`

Member 5: Programs & Outcomes

Focus Areas:

- `resources/js/Pages/Programs/` - Program Pages
- `resources/js/Pages/Outcomes/` - Outcome Pages
- `app/Http/Controllers/Api/ProgramController.php` (with Member 1)
- `app/Http/Controllers/Api/OutcomeController.php` (with Member 1)

Key Files to Create:

- `resources/js/Pages/Programs/Index.jsx`
- `resources/js/Pages/Programs/Create.jsx`
- `resources/js/Pages/Outcomes/Index.jsx`
- `resources/js/Pages/Outcomes/Create.jsx`

Useful Commands

Laravel Commands

```
# Create controller
php artisan make:controller Api/ProgramController --api

# Create model with migration
php artisan make:model Program -m

# Create form request
php artisan make:request StoreProgramRequest

# Create API resource
php artisan make:resource ProgramResource

# Create policy
php artisan make:policy ProgramPolicy --model=Program

# Clear cache
php artisan cache:clear
php artisan config:clear
php artisan route:clear
```

React/Inertia Commands

```
# Create new page component
# (Manual creation in resources/js/Pages/)

# Build for production
npm run build

# Watch for changes
npm run dev
```

Common Issues & Solutions

Issue: "Class not found" errors

Solution:

```
composer dump-autoload
```

Issue: Database connection errors

Solution:

1. Check `.env` file configuration
2. Ensure database server is running
3. Verify database credentials

Issue: Vite build errors

Solution:

```
npm install
npm run dev
```

Issue: Permission errors on storage/logs

Solution:

```
chmod -R 775 storage
chmod -R 775 bootstrap/cache
```

Learning Resources

Laravel

- Laravel Documentation
- Laravel Eloquent Relationships
- Laravel API Resources

React/Inertia

- Inertia.js Documentation
- React Documentation
- Tailwind CSS Documentation

Testing

- PHPUnit Documentation
- Laravel Testing

Next Steps

1. **Week 1**: Focus on basic CRUD operations
2. **Week 2**: Implement entity relationships
3. **Week 3**: Add authentication and advanced features
4. **Week 4**: Testing and deployment preparation

Remember to communicate regularly with your team members and don't hesitate to ask for help when needed!