Development Guide

This guide covers the development setup and workflow for the CapitalFlow Portal.

🚀 Getting Started

Prerequisites

- Node.js 18+
- PostgreSQL 13+
- Yarn package manager
- Git

Local Development Setup

1. **Clone the repository**
   `bash`
   `git clone https://github.com/your-username/capitalflow-portal.git`
   `cd capitalflow-portal`

2. **Install dependencies**
   `bash`
   `cd app`
   `yarn install`

3. **Set up environment variables**
   `bash`
   `cp .env.example .env`

Edit `.env` with your local configuration:
`env`
`DATABASE_URL="postgresql://username:password@localhost:5432/capitalflow_dev"`
`NEXTAUTH_URL="http://localhost:3000"`
`NEXTAUTH_SECRET="your-development-secret-key"`

1. **Set up the database**
   ```bash
   # Generate Prisma client
   npx prisma generate

# Push schema to database
npx prisma db push

# Seed with sample data
npx prisma db seed
```

1. **Start the development server**
   `bash`
   `yarn dev`

The application will be available at `http://localhost:3000`

🏗️ Project Structure

```
capitalflow-portal/
├── app/                    # Next.js application
│   ├── api/               # API routes
│   ├── components/        # React components
│   ├── lib/               # Utility functions
│   ├── prisma/            # Database schema
│   └── scripts/           # Database scripts
├── docs/                  # Documentation
├── .github/               # GitHub configuration
├── docker-compose.yml     # Docker configuration
└── README.md             # Project documentation
```

🔧 Development Workflow

Code Style and Formatting

1. **ESLint Configuration**
   ```
   bash
   yarn lint          # Check for linting errors
   yarn lint:fix      # Fix auto-fixable linting errors
   ```

2. **Prettier Configuration**
   ```
   bash
   yarn format        # Format code
   yarn format:check  # Check formatting
   ```

3. **TypeScript**
   ```
   bash
   yarn type-check    # Type checking
   ```

Database Development

1. **Schema Changes**
   ```
   bash
   # Edit prisma/schema.prisma
   npx prisma db push    # Push changes to database
   npx prisma generate   # Regenerate Prisma client
   ```

2. **Database Management**
   ```
   bash
   npx prisma studio     # Open Prisma Studio
   npx prisma db seed    # Seed database
   npx prisma db reset   # Reset database
   ```

Git Workflow

1. **Create a feature branch**
   ```
   bash
   git checkout -b feature/feature-name
   ```

2. **Make changes and commit**
   ```
   bash
   ```

```
   git add .
   git commit -m "feat: add new feature"
```

3. **Push and create PR**
   `bash`
   ```
   git push origin feature/feature-name
   ```

Testing

1. **Manual Testing**
   - Test with the seeded user: `john@doe.com` / `johndoe123`
   - Test all major features and user flows
   - Test responsive design on different devices

2. **API Testing**
   ```bash
   # Test health endpoint
   curl http://localhost:3000/api/health

# Test authentication endpoints
curl -X POST http://localhost:3000/api/auth/signin \
-H “Content-Type: application/json” \
-d ‘{“email”: “john@doe.com”, “password”: “johndoe123”}’
```

🎨 UI/UX Development

Component Guidelines

1. **Use Radix UI components** for accessibility
2. **Follow Tailwind CSS** patterns for styling
3. **Implement proper error handling** in components
4. **Add loading states** for async operations
5. **Use TypeScript** for all components

Design System

1. **Colors**
   - Primary: Blue (#3B82F6)
   - Secondary: Gray (#6B7280)
   - Success: Green (#10B981)
   - Warning: Yellow (#F59E0B)
   - Error: Red (#EF4444)

2. **Typography**
   - Headings: Inter font family
   - Body: Inter font family
   - Monospace: JetBrains Mono

3. **Spacing**
   - Use Tailwind’s spacing scale
   - Consistent padding and margins
   - Proper component spacing

Responsive Design

1. **Mobile-first approach**
2. **Breakpoints**:
   - `sm` : 640px
   - `md` : 768px
   - `lg` : 1024px
   - `xl` : 1280px
   - `2xl` : 1536px

🔍 Debugging

Common Issues

1. **Database Connection Issues**
   ```bash
   # Check database connection
   psql -h localhost -U username -d capitalflow_dev

# Check Prisma connection
npx prisma db pull
```

1. **Next.js Build Issues**
   ```bash
   # Clear Next.js cache
   rm -rf .next

# Rebuild
yarn build
```

1. **TypeScript Errors**
   ```bash
   # Clear TypeScript cache
   rm -rf .next/cache

# Check types
yarn type-check
```

Development Tools

1. **Prisma Studio**
   `bash
   npx prisma studio`

2. **Next.js Dev Tools**
   - React Developer Tools
   - Next.js DevTools

3. **Database Tools**
   - pgAdmin
   - DBeaver
   - TablePlus

📱 Mobile Development

Testing on Mobile

1. **Local Network Testing**
   ```bash
   # Start dev server accessible on network
   yarn dev --hostname 0.0.0.0
   ```

2. **Mobile-specific Features**
   - Touch interactions
   - Responsive layouts
   - Performance optimization

🚀 Performance Optimization

Code Optimization

1. **Bundle Analysis**
   ```bash
   yarn build:analyze
   ```

2. **Image Optimization**
   - Use Next.js Image component
   - Proper image sizing
   - WebP format support

3. **Code Splitting**
   - Dynamic imports
   - Route-based splitting
   - Component-level splitting

Database Optimization

1. **Query Optimization**
   ```sql
   -- Add indexes for frequently queried columns
   CREATE INDEX idx_user_email ON users(email);
   CREATE INDEX idx_transaction_date ON transactions(transaction_date);
   ```

2. **Connection Pooling**
   ```env
   DATABASE_URL="postgresql://user:password@localhost:5432/db?pgbouncer=true"
   ```

🔒 Security in Development

Security Practices

1. **Environment Variables**
   - Never commit `.env` files
   - Use strong, unique secrets
   - Rotate secrets regularly

2. **Input Validation**
   - Validate all user inputs
   - Use Zod for schema validation
   - Sanitize data before database operations

3. **Authentication**
   - Test authentication flows
   - Verify session management
   - Check authorization levels

Security Testing

1. **OWASP Top 10**
   - Test for common vulnerabilities
   - SQL injection testing
   - XSS testing

2. **Dependency Security**
   ```bash
   yarn audit
   ```

🧪 Testing Strategy

Unit Testing (Future)

1. **Jest Configuration**
   ```bash
   # Future: Set up Jest
   yarn add --dev jest @testing-library/react
   ```

2. **Component Testing**
   ```javascript
   // Example test structure
   describe('Dashboard Component', () => {
     it('renders correctly', () => {
       // Test implementation
     });
   });
   ```

Integration Testing

1. **API Testing**
   ```bash
   # Test API endpoints
   curl http://localhost:3000/api/health
   ```

2. **Database Testing**
   ```bash
   # Test database operations
   npx prisma db seed
   ```

📊 Monitoring in Development

Application Monitoring

1. **Console Logging**
   ```
   javascript
   ```
   ```
   console.log('Development info:', data);
   ```

2. **Performance Monitoring**
   - React DevTools Profiler
   - Next.js DevTools
   - Browser Performance tab

Database Monitoring

1. **Query Analysis**
   ```
   sql
   ```
   ```
   EXPLAIN ANALYZE SELECT * FROM users WHERE email = 'user@example.com';
   ```

2. **Connection Monitoring**
   ```
   sql
   ```
   ```
   SELECT * FROM pg_stat_activity;
   ```

🤝 Contributing

Code Review Process

1. **Self Review**
   - Check code quality
   - Test functionality
   - Review documentation

2. **Peer Review**
   - Create pull request
   - Address feedback
   - Ensure tests pass

Documentation

1. **Code Documentation**
   - JSDoc comments
   - README updates
   - API documentation

2. **Change Documentation**
   - Update CHANGELOG.md
   - Document breaking changes
   - Update migration guides

📚 Learning Resources

Next.js Resources

- Next.js Documentation (https://nextjs.org/docs)
- Next.js Examples (https://github.com/vercel/next.js/tree/main/examples)

- Next.js Best Practices (https://nextjs.org/docs/basic-features/font-optimization)

React Resources

- React Documentation (https://reactjs.org/docs)
- React Hooks (https://reactjs.org/docs/hooks-intro.html)
- React Patterns (https://reactpatterns.com/)

TypeScript Resources

- TypeScript Handbook (https://www.typescriptlang.org/docs)
- TypeScript with React (https://react-typescript-cheatsheet.netlify.app/)

Database Resources

- Prisma Documentation (https://www.prisma.io/docs)
- PostgreSQL Documentation (https://www.postgresql.org/docs/)

Happy coding! 🚀