7-Day Frappe & ERPNext Study Plan + Development Learning Strategy

Week 1: Intensive Study Phase (7 Days)

Day 1: Foundation & Environment Setup

Morning (3-4 hours)

- Install Frappe/ERPNext development environment
- Set up VS Code with relevant extensions
- Understand the Frappe architecture overview
- Learn about the MVC pattern in Frappe

Afternoon (3-4 hours)

- Study Frappe's folder structure
- Understand DocTypes concept
- Create your first custom DocType
- Practice with basic field types

Evening (1-2 hours)

- Read Frappe documentation basics
- Watch introductory videos on Frappe framework

Day 2: DocTypes & Database Layer

Morning (3-4 hours)

- Deep dive into DocType creation
- Understanding field types and their properties
- Learn about naming series and auto-naming
- Practice creating related DocTypes

Afternoon (3-4 hours)

- Database relationships (Link, Table, Child Table)
- Permissions and user roles
- Custom fields and custom forms

- Data import/export basics

Evening (1-2 hours)

- Practice exercises with customer/supplier DocTypes
- Review ERPNext's core DocTypes structure

Day 3: Server-Side Scripting

Morning (3-4 hours)

- Python scripting in Frappe
- Controller methods (validate, before_save, after_insert)
- Server-side API creation
- Understanding frappe.db methods

Afternoon (3-4 hours)

- Hooks and fixtures
- Background jobs and scheduling
- Email integration and notifications
- Error handling and logging

Evening (1-2 hours)

- Practice creating business logic
- Work on validation scripts

Day 4: Client-Side Development

Morning (3-4 hours)

- JavaScript in Frappe framework
- Client-side scripting for forms
- Custom buttons and actions
- Form events and triggers

Afternoon (3-4 hours)

- Page and Report development
- Dashboard creation

- Custom print formats
- Understanding Jinja templating

Evening (1-2 hours)

- Practice frontend customizations
- Experiment with form layouts

Day 5: ERPNext Modules Deep Dive

Morning (3-4 hours)

- Accounting module structure
- Sales and Purchase cycles
- Stock management concepts
- Understanding ERPNext's business logic

Afternoon (3-4 hours)

- Manufacturing module
- HR and Payroll basics
- Project management features
- CRM functionalities

Evening (1-2 hours)

- Explore ERPNext customization options
- Study existing ERPNext apps on GitHub

Day 6: Advanced Features & Integration

Morning (3-4 hours)

- REST API development and consumption
- Third-party integrations
- Custom apps development
- App structure and packaging

Afternoon (3-4 hours)

- Workflow automation

- Custom reports and queries
- Performance optimization basics
- Security best practices

Evening (1-2 hours)

- Practice API calls
- Review integration patterns

Day 7: Testing & Deployment

Morning (3-4 hours)

- Unit testing in Frappe
- Bench commands and management
- Site management and multi-tenancy
- Backup and restore procedures

Afternoon (3-4 hours)

- Production deployment concepts
- Performance monitoring
- Debugging techniques
- Code review best practices

Evening (1-2 hours)

- Create a mini project combining learned concepts
- Prepare development environment for real project

Development Phase: Learning While Building

Pre-Development Setup

- Set up version control (Git) with proper branching strategy
- Create development, staging, and production environments
- Establish code review process
- Set up documentation structure

Learning-While-Developing Strategy

Week 1-2: Project Foundation

Focus Areas:

- Apply DocType design patterns from study week
- Implement core business logic
- Set up proper project structure

Learning Approach:

- Start with simple features and gradually increase complexity
- Document every custom solution you create
- Create reusable components and utilities
- Regular code reviews with senior developers

Week 3-4: Feature Development

Focus Areas:

- Implement complex business workflows
- Create custom reports and dashboards
- Integrate with external systems

Learning Approach:

- Research specific solutions as you encounter challenges
- Contribute to Frappe/ERPNext community forums
- Study ERPNext source code for similar implementations
- Maintain a learning journal of solutions found

Week 5-6: Optimization & Advanced Features

Focus Areas:

- Performance optimization
- Advanced customizations
- Mobile responsiveness
- User experience improvements

Learning Approach:

- Profile and optimize slow queries
- Learn advanced JavaScript/Python patterns
- Study UX best practices for ERP systems
- Implement progressive enhancements

Ongoing Development Best Practices

Daily Learning Habits:

- Spend 30 minutes reading Frappe/ERPNext forums
- Keep ERPNext documentation bookmarked for quick reference
- Follow ERPNext GitHub repository for updates
- Practice explaining complex concepts to team members

Weekly Learning Goals:

- Learn one new advanced feature per week
- Contribute to open-source discussions
- Refactor and improve existing code
- Update documentation and create tutorials

Problem-Solving Approach:

1. Check official documentation first
2. Search community forums and GitHub issues
3. Examine ERPNext source code for similar patterns
4. Ask specific questions on forums with code examples
5. Document solutions for future reference

Resource Recommendations

Essential Documentation:

- Frappe Framework Documentation
- ERPNext User Manual
- ERPNext Developer Documentation
- Frappe School tutorials

Community Resources:

- ERPNext Community Forum
- Frappe Framework Discuss
- ERPNext GitHub repository
- Frappe School YouTube channel

Tools for Development:

- VS Code with Python and JavaScript extensions
- Git for version control
- Postman for API testing
- Browser developer tools
- Frappe Bench for site management

Success Metrics

After Study Week:

- Ability to create custom DocTypes independently
- Understanding of ERPNext business processes
- Comfortable with basic Python/JavaScript in Frappe
- Can navigate and understand ERPNext codebase

After Development Phase:

- Successfully delivered project features
- Contributed to code quality improvements
- Mentored other team members
- Created reusable components and documentation

Emergency Learning Protocol

When stuck during development:

1. Time-box problem-solving (max 2 hours)
2. Document the issue clearly
3. Search for similar problems in community
4. Ask for help with specific code examples
5. Schedule knowledge sharing session with team

6. Update personal knowledge base with solution

This plan balances intensive upfront learning with practical application, ensuring you build both theoretical knowledge and real-world development skills.