FastAPI Learning Roadmap

Phase 1: Foundations (Week 1-2)

Python Basics Review

- Async/await fundamentals
- Type hints and annotations
- Decorators
- Context managers
- List/dict comprehensions

FastAPI Core Concepts

- Install FastAPI and Uvicorn
- Your first API endpoint
- Path parameters and query parameters
- Request body with Pydantic models
- Response models and status codes
- Automatic interactive docs (Swagger UI)

Practice Project

Build a simple Todo API with in-memory storage (list/dict)

Phase 2: Database Integration (Week 3-4)

SQLAlchemy & Databases

- SQLAlchemy Core vs ORM
- Defining models and relationships
- CRUD operations
- Database migrations with Alembic
- Async SQLAlchemy (important!)

Database Connections

- Connection pooling

- Database sessions and dependency injection
- PostgreSQL setup and connection
- Environment variables with python-decouple or pydantic-settings

Practice Project

Build a Blog API: users, posts, comments with relationships

Phase 3: Authentication & Security (Week 5-6)

Auth Fundamentals

- Password hashing with passlib/bcrypt
- JWT tokens (access + refresh)
- OAuth2 with Password Bearer
- Dependency injection for auth
- Protected routes

Security Best Practices

- CORS configuration for React frontend
- Rate limiting
- Input validation and sanitization
- SQL injection prevention (SQLAlchemy handles this)
- Environment secrets management

Practice Project

Add authentication to your Blog API with user registration/login

---

Phase 4: Advanced Features (Week 7-8)

File Handling

- File uploads
- Image processing (Pillow)
- Serving static files
- Streaming responses

Background Tasks

- Background tasks in FastAPI
- Celery for heavy jobs
- Redis for task queues
- WebSockets for real-time features

API Design Patterns

- Pagination
- Filtering and sorting
- API versioning
- Error handling and custom exceptions
- Response model optimization (exclude fields)

Practice Project

E-commerce API: products, cart, orders, image uploads, email notifications

---

Phase 5: Testing & Quality (Week 9-10)

Testing

- pytest basics
- Testing FastAPI endpoints with TestClient
- Fixtures and mocking
- Database testing with test database
- Test coverage with pytest-cov

Code Quality

- Type checking with mypy
- Linting with ruff or pylint
- Code formatting with black
- Pre-commit hooks

Practice Project

Write comprehensive tests for your E-commerce API (80%+ coverage)

Phase 6: Performance & Optimization (Week 11-12)

Caching

- Redis for caching
- Cache strategies (cache-aside, write-through)
- Cache invalidation patterns
- FastAPI's built-in response caching

Performance Optimization

- Database query optimization (N+1 queries)
- Indexes and explain analyze
- Async database drivers (asyncpg for PostgreSQL)
- Connection pooling tuning
- Profiling with py-spy or cProfile

Monitoring

- Logging best practices (structlog)
- Application metrics (Prometheus)
- Error tracking (Sentry)
- Request tracing

Practice Project

Optimize your E-commerce API to handle 10k+ requests/second

Phase 7: Deployment & DevOps (Week 13-14)

Containerization

- Docker basics
- Dockerfile for FastAPI
- Docker Compose for local development
- Multi-stage builds

Deployment Options

- Deploy to Railway (easiest)

- Deploy to AWS (EC2, ECS, or Lambda)
- Deploy to DigitalOcean App Platform
- Deploy to Google Cloud Run
- Deploy to Render

Production Considerations

- Gunicorn + Uvicorn workers
- Environment-based configuration
- Database backups
- SSL/TLS certificates (Let's Encrypt)
- Reverse proxy with Nginx (optional)

CI/CD

- GitHub Actions for testing
- Automated deployments
- Database migrations in CI/CD

Practice Project

Deploy your E-commerce API to production with CI/CD

Phase 8: Real-World Integration (Week 15-16)

Third-Party Integrations

- Payment processing (Stripe)
- Email services (SendGrid, AWS SES)
- Cloud storage (AWS S3)
- SMS notifications (Twilio)

React Frontend Integration

- CORS setup for React app
- API client in React (axios/fetch)
- JWT storage and refresh logic
- Error handling in React
- File uploads from React

Microservices Basics

- When to split services
- Service-to-service communication
- API Gateway pattern
- Message queues (RabbitMQ/Redis)

Practice Project

Full-stack app: React frontend + FastAPI backend with payment integration

---

Continuous Learning

Advanced Topics

- GraphQL with Strawberry
- gRPC with FastAPI
- Server-Sent Events (SSE)
- WebSockets for chat/real-time
- Event-driven architecture
- Domain-Driven Design patterns

Resources

- Official FastAPI docs (incredibly good!)
- FastAPI GitHub discussions
- Real Python tutorials
- testdriven.io FastAPI course
- Build real projects and contribute to open source

Milestone Projects

Beginner

Simple Todo API with authentication

Intermediate

Blog platform with comments, likes, image uploads

Advanced

E-commerce platform with payments, inventory, notifications

Expert

Multi-tenant SaaS application with usage-based billing

Key Success Metrics

By the end of this roadmap, you should be able to:

- Build production-ready APIs in 1-2 weeks
- Handle 10k+ concurrent users
- Implement secure authentication
- Write comprehensive tests
- Deploy to cloud platforms
- Integrate with React frontends
- Debug and optimize performance issues
- Make architectural decisions confidently

Estimated total time: 3-4 months with consistent daily practice (2-3 hours/day)