# 🗓️ InterPay: Daily Development Plan (Starting Tomorrow)

## 📅 \*\*Phase 1: Foundation & Infrastructure (Weeks 1-3)\*\*

### \*\*Week 1: Project Setup & Basic Architecture\*\* (Dec 9 - Dec 15, 2024)

\*\*Day 1 - Mon, Dec 9\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: Microservices vs Monolith, Spring Boot basics

- \*\*Development\*\*:

- [ ] Create GitHub repository `interpay-system`

- [ ] Initialize project structure with 12 microservice folders

- [ ] Set up Spring Boot 3.x for api-gateway service

- [ ] Create basic "Hello World" endpoint in gateway

- \*\*Success Check\*\*: `curl http://localhost:8080` returns response

\*\*Day 2 - Tue, Dec 10\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: Service Discovery patterns, Eureka Server

- \*\*Development\*\*:

- [ ] Implement Eureka Server (service-discovery)

- [ ] Register api-gateway with Eureka

- [ ] Test service registration in Eureka dashboard

- [ ] Add health check endpoints

- \*\*Success Check\*\*: Eureka dashboard shows registered services

\*\*Day 3 - Wed, Dec 11\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: API Gateway patterns, routing configuration

- \*\*Development\*\*:

- [ ] Configure Spring Cloud Gateway routes

- [ ] Implement path-based routing (`/api/users/\*\*` → user-service)

- [ ] Add load balancing between service instances

- [ ] Create custom filters for logging

- \*\*Success Check\*\*: Gateway routes requests to correct services

\*\*Day 4 - Thu, Dec 12\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: Centralized configuration management

- \*\*Development\*\*:

- [ ] Set up Spring Cloud Config Server

- [ ] Create configuration files for each service

- [ ] Implement dynamic configuration refresh

- [ ] Test configuration changes without restart

- \*\*Success Check\*\*: Services fetch config from central server

\*\*Day 5 - Fri, Dec 13\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: Docker fundamentals, containerization benefits

- \*\*Development\*\*:

- [ ] Create Dockerfile for api-gateway

- [ ] Dockerize service-discovery and config-server

- [ ] Set up Docker network for inter-service communication

- [ ] Test services running in containers

- \*\*Success Check\*\*: All services run via `docker-compose up`

\*\*Day 6 - Sat, Dec 14\*\* (4h dev)

- \*\*Development\*\*:

- [ ] Create Dockerfiles for all 12 services

- [ ] Set up docker-compose.yml with all services

- [ ] Configure environment variables

- [ ] Test complete system in Docker

- \*\*Success Check\*\*: Full system runs with single `docker-compose up`

\*\*Day 7 - Sun, Dec 15\*\* (4h dev)

- \*\*Weekly Review & Polish\*\*:

- [ ] Fix any integration issues

- [ ] Document setup instructions in README.md

- [ ] Create startup scripts

- [ ] Test complete system startup/shutdown

- \*\*Week 1 Deliverable\*\*: ✅ Full microservices skeleton running in Docker

---

### \*\*Week 2: Database Design & User Management\*\* (Dec 16 - Dec 22, 2024)

\*\*Day 8 - Mon, Dec 16\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: Database-per-service pattern, PostgreSQL basics

- \*\*Development\*\*:

- [ ] Design user-service database schema

- [ ] Set up PostgreSQL container for user-service

- [ ] Implement User entity with JPA

- [ ] Create basic CRUD operations for User

- \*\*Success Check\*\*: User records can be created/read from database

\*\*Day 9 - Tue, Dec 17\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: JWT authentication, Spring Security

- \*\*Development\*\*:

- [ ] Implement user registration endpoint `POST /api/users/register`

- [ ] Add password encryption with BCrypt

- [ ] Create login endpoint `POST /api/auth/login`

- [ ] Generate JWT tokens on successful login

- \*\*Success Check\*\*: Users can register and receive JWT tokens

\*\*Day 10 - Wed, Dec 18\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: JWT validation, security filters

- \*\*Development\*\*:

- [ ] Implement JWT validation filter in api-gateway

- [ ] Add role-based access control (USER, ADMIN)

- [ ] Create protected endpoints requiring authentication

- [ ] Implement token refresh mechanism

- \*\*Success Check\*\*: Protected endpoints reject requests without valid JWT

\*\*Day 11 - Thu, Dec 19\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: Database relationships, wallet design patterns

- \*\*Development\*\*:

- [ ] Design wallet-service database schema

- [ ] Implement Wallet entity (id, userId, balance, currency)

- [ ] Create wallet creation on user registration

- [ ] Add balance inquiry endpoint

- \*\*Success Check\*\*: Each user has an associated wallet

\*\*Day 12 - Fri, Dec 20\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: ACID properties, transaction management

- \*\*Development\*\*:

- [ ] Implement wallet balance update operations

- [ ] Add transaction history table

- [ ] Create audit logging for balance changes

- [ ] Implement concurrent access handling

- \*\*Success Check\*\*: Balance updates are atomic and consistent

\*\*Day 13 - Sat, Dec 21\*\* (4h dev)

- \*\*Development\*\*:

- [ ] Set up databases for all services

- [ ] Implement database migration scripts

- [ ] Add connection pooling configuration

- [ ] Create database health checks

- \*\*Success Check\*\*: All services connected to their databases

\*\*Day 14 - Sun, Dec 22\*\* (4h dev)

- \*\*Weekly Review & Polish\*\*:

- [ ] Test complete user registration flow

- [ ] Validate JWT security across all services

- [ ] Create database backup strategies

- [ ] Document API endpoints in Postman

- \*\*Week 2 Deliverable\*\*: ✅ User management with authentication & basic wallets

---

### \*\*Week 3: DevOps Foundation\*\* (Dec 23 - Dec 29, 2024)

\*\*Day 15 - Mon, Dec 23\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: Docker networking, multi-container apps

- \*\*Development\*\*:

- [ ] Optimize Dockerfiles for smaller image sizes

- [ ] Implement Docker health checks

- [ ] Configure container resource limits

- [ ] Set up container logging

- \*\*Success Check\*\*: Containers are optimized and monitored

\*\*Day 16 - Tue, Dec 24\*\* (4h dev + 1h learning) \*Christmas Eve\*

- \*\*Learning\*\*: Docker Compose for development

- \*\*Development\*\*:

- [ ] Create development docker-compose with hot reload

- [ ] Set up database initialization scripts

- [ ] Configure service dependencies

- [ ] Add utility containers (adminer, etc.)

- \*\*Success Check\*\*: `docker-compose up` starts complete development environment

\*\*Day 17 - Wed, Dec 25\*\* (4h dev) \*Christmas Day\*

- \*\*Development\*\*:

- [ ] Learn basic Kubernetes concepts

- [ ] Create Kubernetes namespace configuration

- [ ] Implement basic pod definitions

- [ ] Set up kubectl configuration

- \*\*Success Check\*\*: Can deploy pods to local Kubernetes

\*\*Day 18 - Thu, Dec 26\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: Kubernetes Services and Deployments

- \*\*Development\*\*:

- [ ] Create Kubernetes deployments for all services

- [ ] Implement Kubernetes services for discovery

- [ ] Configure liveness and readiness probes

- [ ] Set up resource requests and limits

- \*\*Success Check\*\*: All services deployed to Kubernetes

\*\*Day 19 - Fri, Dec 27\*\* (4h dev + 1h learning)

- \*\*Learning\*\*: Basic monitoring with Spring Boot Actuator

- \*\*Development\*\*:

- [ ] Add Spring Boot Actuator to all services

- [ ] Implement custom health indicators

- [ ] Set up metrics collection

- [ ] Create basic dashboard with Grafana

- \*\*Success Check\*\*: Can monitor service health and metrics

\*\*Day 20 - Sat, Dec 28\*\* (4h dev)

- \*\*Development\*\*:

- [ ] Set up GitHub Actions for CI

- [ ] Create build and test pipeline

- [ ] Implement automated Docker builds

- [ ] Add deployment to Kubernetes on master merge

- \*\*Success Check\*\*: CI/CD pipeline builds and deploys on git push

\*\*Day 21 - Sun, Dec 29\*\* (4h dev)

- \*\*Weekly Review & Polish\*\*:

- [ ] Test complete local Kubernetes deployment

- [ ] Verify CI/CD pipeline works end-to-end

- [ ] Document deployment procedures

- [ ] Create troubleshooting guide

- \*\*Week 3 Deliverable\*\*: ✅ Full DevOps pipeline with Kubernetes deployment

---

## 🎯 \*\*End of Phase 1 Review\*\* (Dec 30, 2024)

\*\*Status Check\*\*:

- [ ] 12 microservices running in Kubernetes

- [ ] User registration and authentication working

- [ ] Basic wallet system implemented

- [ ] Full CI/CD pipeline operational

- [ ] Monitoring and health checks in place

\*\*Phase 1 Complete!\*\* ✅ You now have a solid foundation. Ready to move to core banking services!

---

## 📋 \*\*Daily Work Template\*\*

### \*\*Daily Structure\*\* (5 hours total):

```

4:00 PM - 4:30 PM | Review yesterday's progress & plan today's tasks

4:30 PM - 6:30 PM | Development Session 1 (2 hours)

6:30 PM - 7:00 PM | Break

7:00 PM - 8:00 PM | Learning Session (1 hour)

8:00 PM - 9:00 PM | Development Session 2 (1 hour)

9:00 PM - 9:30 PM | Document progress & prepare for tomorrow

```

### \*\*Daily Success Criteria\*\*:

- [ ] Code committed to GitHub

- [ ] Learning notes documented

- [ ] Tomorrow's tasks defined

- [ ] Any blockers identified and addressed

### \*\*Progress Tracking\*\*:

- Use GitHub Projects for task management

- Maintain daily development journal

- Weekly demo videos (record 5-min walkthrough)

- Keep API documentation updated in Postman

\*\*Ready to start tomorrow?\*\* This daily plan will ensure consistent progress toward your $100K FinTech goal! 🚀

Would you like me to continue with the daily breakdown for Phase 2?