Strategic Expansion: The Tax Calc Carrier App

Integrating with Magento and Salesforce Commerce Cloud

Executive summary

Market Opportunity

- Magento dominates mid-market segment with high customization needs
- SFCC leads enterprise B2C with advanced personalization capabilities
- Both platforms have merchants actively seeking cross-border solutions
- Limited competition in integrated shipping + tax + duty calculation space

Expanding Horizons

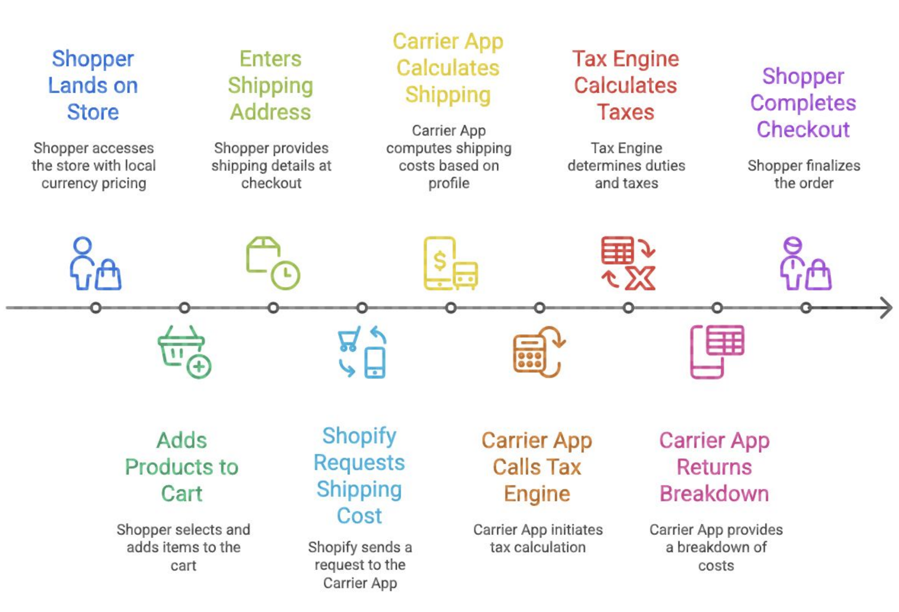
The Goal: Integration with 2 New Enterprise Platforms i.e. Magento & SFCC.
The Core Promise: <1 Week Merchant Onboarding.
The Target: 100+ New Merchant Activations.
Seamless Cross Border Checkout: Our integration ensures a smooth, localized experience for international shoppers with accurate taxes and optimized payments.

Strategic TaxCalcjective:

Expand Tax Calc's Carrier App to Magento and SFCC platforms, targeting mid-market and enterprise merchants with seamless cross-border shipping, tax, and duty calculations while maintaining our signature <1 week onboarding experience.

AS IS Online Shopping Process

e-Commerce Checkout Process



Shopper Lands on Store

Shopper accesses the store with local currency pricing

Adds Products to Cart

Shopper selects and adds items to the cart

Enters Shipping Address

Shopper provides shipping details at checkout

Shopify Requests Shipping Cost

Shopify sends a request to the Carrier App

Carrier App Calculates Shipping

Carrier App computes shipping costs based on profile

Carrier App Calls Tax Engine

Carrier App initiates tax calculation

Tax Engine Calculates Taxes

Tax Engine determines duties and taxes

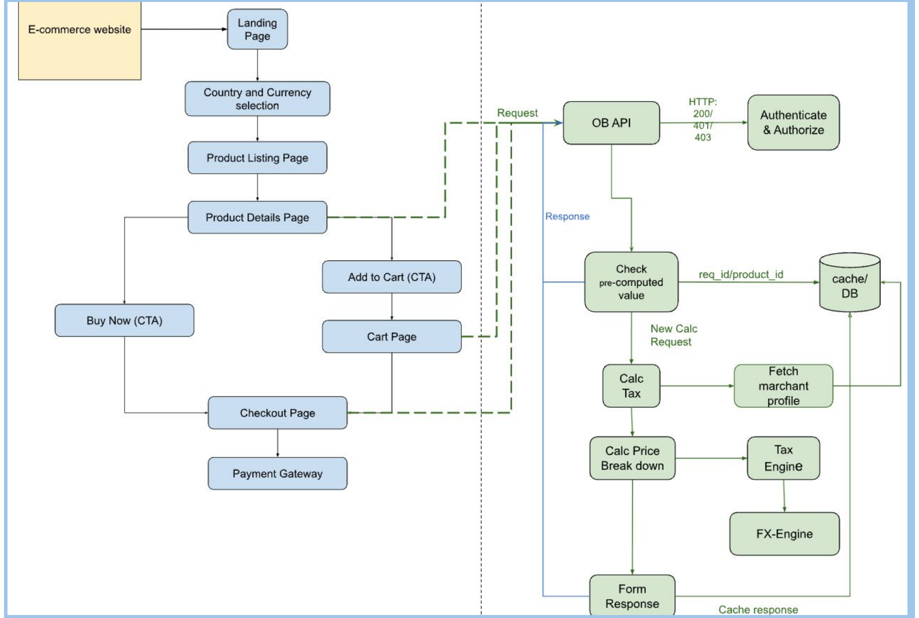
Carrier App Returns Breakdown

Carrier App provides a breakdown of costs

Shopper Completes Checkout

Shopper finalizes the order

System Architecture



Two Steps Integration Pathway

Basic Integration

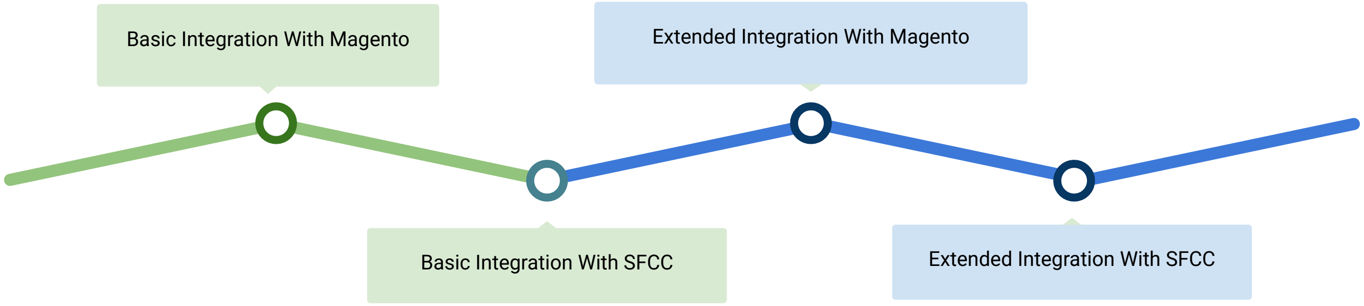
The essential package for enabling cross-border sales with full cost transparency.

✓Dynamic Shipping Calculations
✓Automated Tax Calculations
✓Accurate Duty Estimations

Extended Integration

A complete solution that includes everything in Basic, plus localized payment processing to maximize conversion.

✓All Basic Features
✓Secure Local Payment Gateway
✓Localized Payment Methods (e.g., Interac, iDEAL)



Basic Integration With Magento

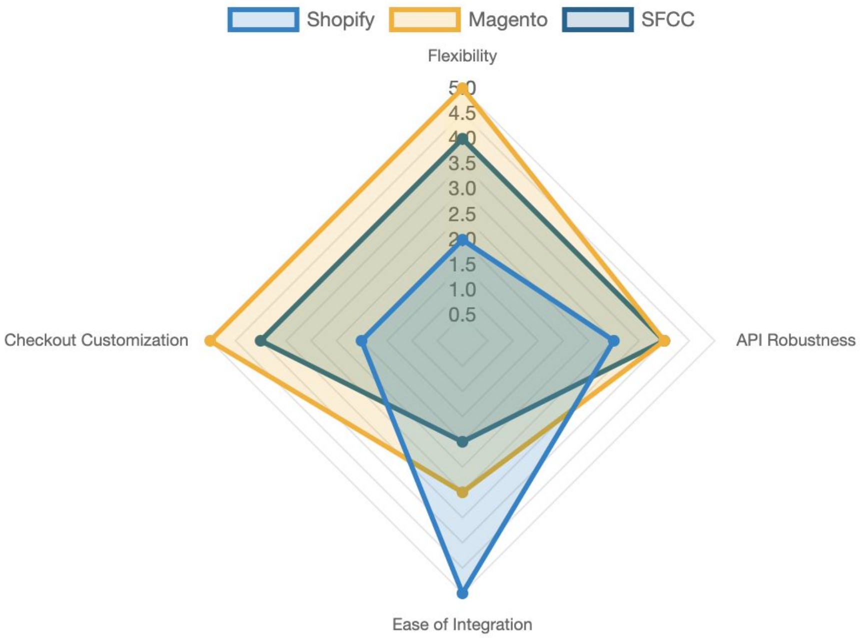
Basic Integration With SFCC

Extended Integration With Magento

Extended Integration With SFCC

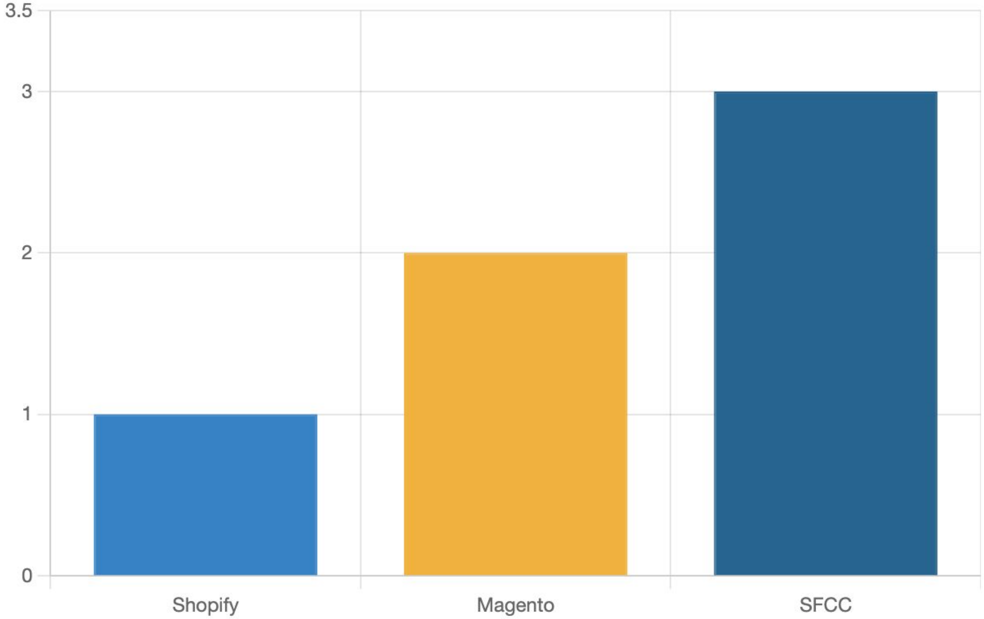
Effort Comparison

Architectural Characteristics



Comparing platforms across key architectural and extensibility dimensions.

Integration Effort Comparison



A qualitative assessment of development effort required for each platform integration.

Key Differences Between Platforms

Characteristic	Shopify	Magento	SFCC
Extensibility & Architecture	Primarily app-based, leveraging a rTaxCalcust API ecosystem for integrations. It's SaaS, so less core code customization is possible. Its checkout.liquid file offers some control but is generally more restrictive. Plugin-based, but limited control	A highly flexible, open-source platform with deep customization via modules, themes, and PHP extensions. Requires custom integration due to its modular architecture and is more complex than app-based platforms like Shopify. Supports both self-hosted and cloud deployment.	Enterprise-grade SaaS platform with a cartridge-based architecture, rTaxCalcust APIs, and customizable storefronts (SFRA/SiteGenesis). Integrations rely on APIs and webhooks, offering full control but less flexibility than Shopify's app model.
Checkout APIs	Calculates shipping rates at checkout using cart and shipping details via a fixed-flow REST/GraphQL Carrier Service API.	Offers rTaxCalcust REST and SOAP APIs for cart, checkout, and order management. Shipping methods and rates are extensible via custom modules or API calls. Checkout is fully customizable via REST or native PHP hooks.	Leverages OCAPI/SCAPI for checkout, using basket and order APIs to connect shipping and tax services. Requires custom cartridges (ISML/OCAPI) or SFRA controllers with OCAPI hooks
Plugins/Middleware	Relies heavily on public and private apps available on its App Store.	Leverages modules (extensions) from its Marketplace or custom-developed. Middleware could be used for complex integrations. Magento may require middleware (e.g., GraphQL) to handle API versioning and data transformation.	Uses "cartridges" for extending functionality, and often requires custom development for complex integrations. SFCC integrations often require middleware for real-time data sync (e.g., via SFCC's Business Manager)
Payment Integration	Native gateway support	Multiple gateway support. Magento's payment gateway APIs require tighter integration with local payment providers (e.g., Interac for Canada) compared to Shopify's standardized payment stack. Additional configuration for payment method mapping in Magento's admin panel.	Commerce API. SFCC's payment processing requires custom cartridges to integrate local payment methods (e.g., BACS for the UK). SFCC's checkout pipeline demands precise hook placement compared to Shopify's more streamlined payment APIs.
Ease of Integration	Fastest (native app store)	Mid-complexity	Higher dev effort (esp. for SFRA)
Middleware Support	Minimal	Strong (e.g., M2E Pro, API Gateways)	Required for most custom logic

Design Principles & Risks

Key Design Principles

Unified Backend: Single API serves all platforms, reducing maintenance overhead
Platform-Native Integration: Leverage each platform's native extension mechanisms
Real-time Processing: Dynamic rate calculation with <2s response time
Fallback Handling: Graceful degradation for API failures

Risk Analysis & Mitigation

High Risk: Platform API Changes

Risk: Magento/SFCC API deprecation breaks integration
Mitigation: Maintain active partnerships, early access programs, and backward compatibility layers

Medium Risk: Development Complexity

Risk: SFCC cartridge complexity delays launch
Mitigation: Start with MVP, dedicated SFCC expertise, phased rollout

Medium Risk: Market Competition

Risk: Competitors launch similar solutions
Mitigation: Superior onboarding experience, feature depth, merchant relationships

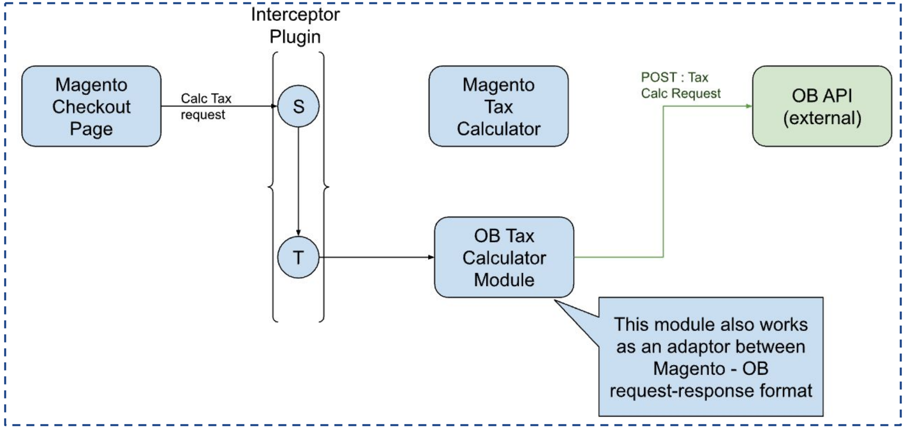
Low Risk: Resource Allocation

Risk: Development resources constrained
Mitigation: Parallel development, shared backend API, contractor support

Key Risk Management Strategies

Early pilot programs with 5 merchants per platform for rapid feedback
Phased rollout approach allowing course correction
Dedicated platform expertise and partner relationships
Comprehensive testing with real merchant scenarios

Magento Integration Strategy



Using Magento Custom Module

Core Integration Components

- Custom Magento module
- Extends Carrier class
- REST API integration
- Multi-store configuration support
- Admin panel configuration

Extended Features

- Local payment methods (Interac, iDEAL)
- Multi-currency support
- Payment gateway integration
- Merchant self-service portal

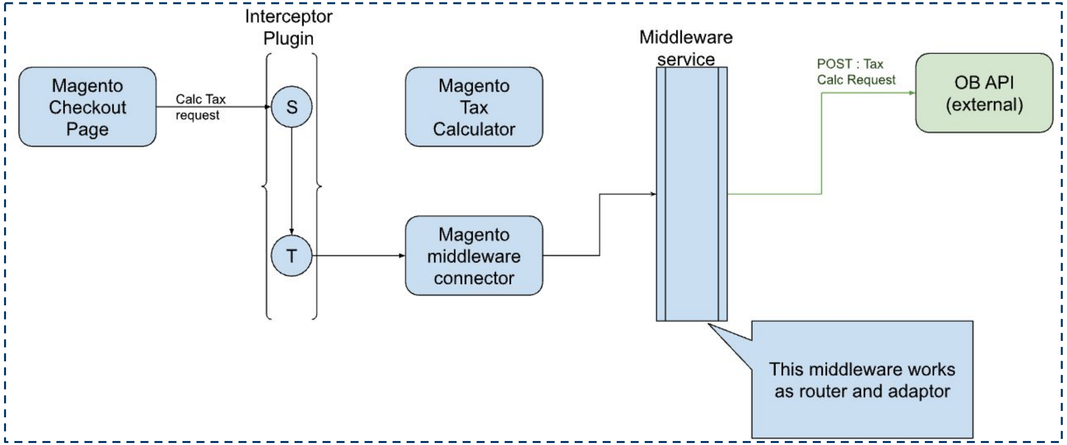
Using Middle Layer

Core Integration Components

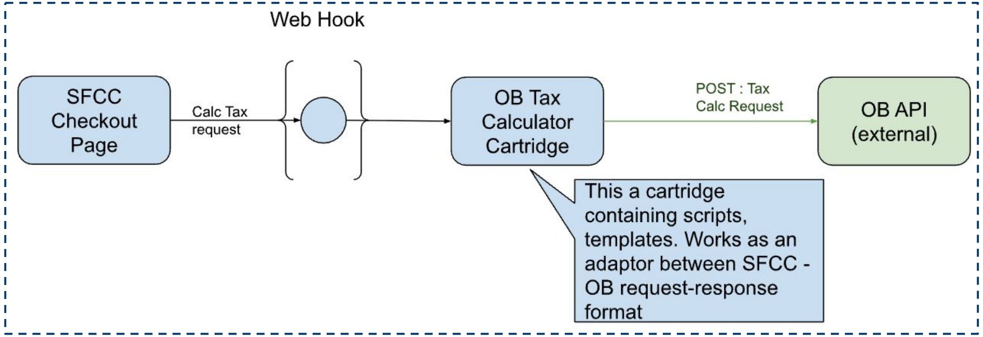
- Same as other integration. But this is more scalable, decoupled architecture that abstracts Magento from external tax service details—enhancing flexibility, security, and performance.

Extended Features

- Same as other but the payment gateway logic can be handled in the Middleware.



SFCC Integration Strategy


Web Hook
SFCC Checkout Page
Calc Tax request
OB Tax Calculator Cartridge
POST : Tax Calc Request
OB API (external)
This a cartridge containing scripts, templates. Works as an adaptor between SFCC - OB request-response format

Using SFCC Custom Cartridge

Core Integration Components

- Custom SFCC cartridge
- OCAPI/SCAPI integration
- Pipeline hook implementation
- Business Manager configuration
- ISML template integration

Extended Features

- Local payment methods (BACS, Sofort)
- Locale-based currency switching
- Commerce API payment processing
- Business Manager configuration

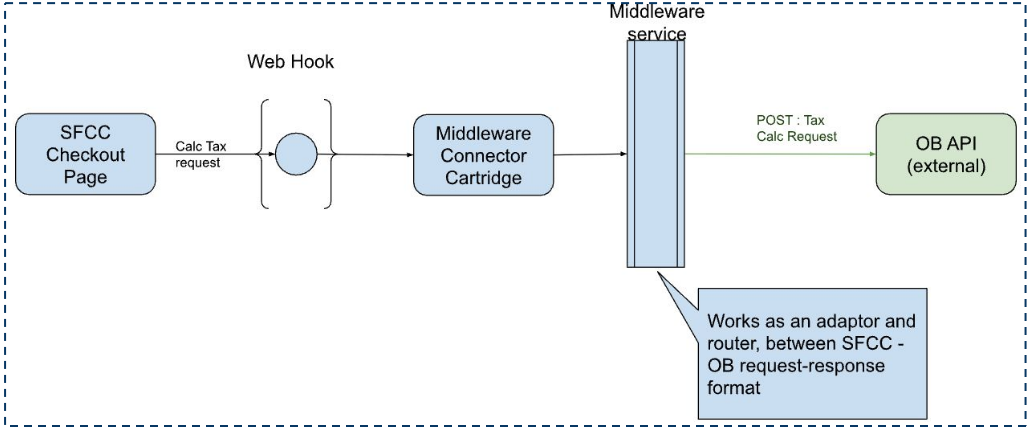
Using Middle Layer

Core Integration Components

- Same as other integration. Adopting a middleware approach ensures a clean separation of concerns by keeping the cartridge lightweight and shifting core logic to the middleware layer. This promotes code reusability across service providers and enables faster time-to-market.

Extended Features

- Same as other but the payment gateway logic can be handled in the Middleware.


Middleware service
Web Hook
SFCC Checkout Page
Calc Tax request
Middleware Connector Cartridge
POST : Tax Calc Request
OB API (external)
Works as an adaptor and router, between SFCC - OB request-response format

Project Roadmap & Timeline

Our phased approach ensures a robust and timely launch for both Magento and SFCC integrations, from development to full production rollout.

Magento Launch Plan (10 Weeks)

Weeks 1-6: Development
Building the custom Magento module and core integration logic.

Weeks 7-8: Beta Testing
Testing with a select group of merchants to gather feedback and ensure stability.

Week 10: Full Launch
Production rollout of the Magento extension to the public.

SFCC Launch Plan (12 Weeks)

Weeks 1-8: Development
Developing the custom SFCC cartridge and integrating with OCAPI.

Weeks 9-10: Beta Testing
Rigorous testing within the SFCC sandbox environment with partner merchants.

Week 12: Full Launch
Deploying the certified SFCC cartridge for general availability.

Go-to-Market Strategy

Partner Channel Strategy

- **Magento:** AdTaxCalce Commerce Marketplace listing
- **SFCC:** Salesforce AppExchange presence
- Solution partner co-marketing programs
- Systems integrator partnerships

Content Marketing

- Platform-specific case studies
- Technical integration guides
- Webinar series for each platform
- Conference presence (Imagine, Connections)

Direct Sales Approach

- Leverage existing merchant relationships
- Platform referral programs
- Targeted outreach to enterprise prospects
- Free pilot program offerings

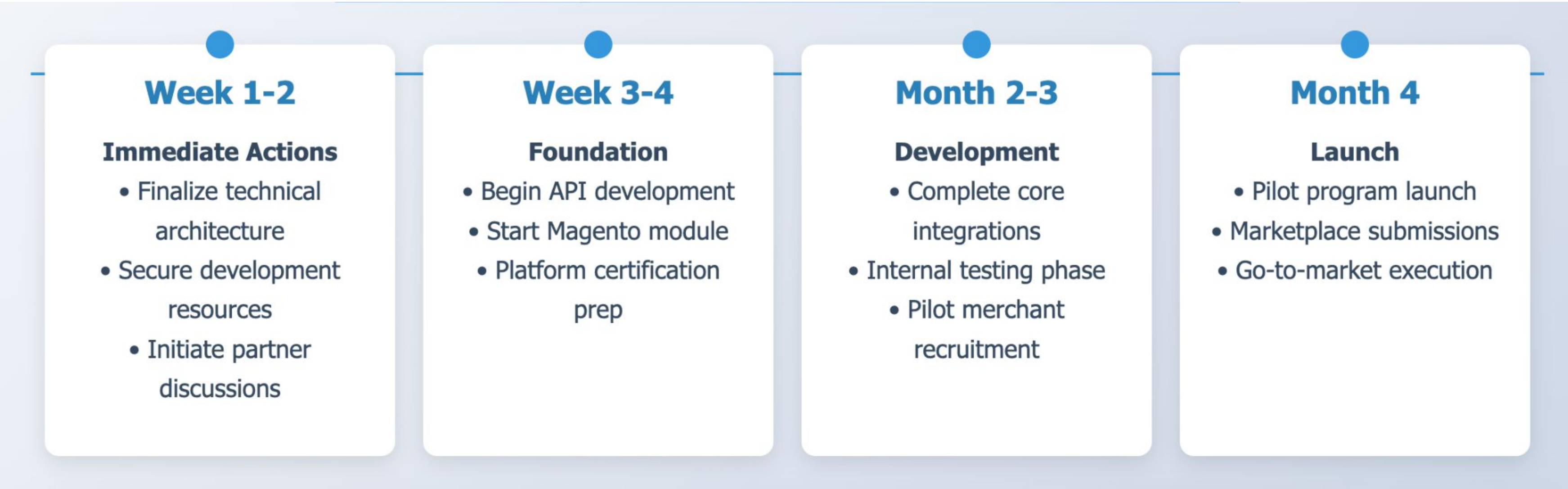
Success Enablement

- Dedicated platform documentation
- Video tutorial libraries
- 24/7 technical support
- Merchant success management

Launch Sequence

Phase 1: Pilot with 5 merchants per platform → **Phase 2:** Partner marketplace launch → **Phase 3:** Full market availability with marketing campaign

Next Steps & Recommendations



Week 1-2

Immediate Actions

- Finalize technical architecture
- Secure development resources
- Initiate partner discussions

Week 3-4

Foundation

- Begin API development
- Start Magento module
- Platform certification prep

Month 2-3

Development

- Complete core integrations
- Internal testing phase
- Pilot merchant recruitment

Month 4

Launch

- Pilot program launch
- Marketplace submissions
- Go-to-market execution

Key Recommendations

- **Prioritize Magento:** Faster time-to-market and larger addressable segment
- **Invest in Platform Expertise:** Hire dedicated Magento/SFCC developers
- **Pilot Program:** Start with 5 merchants per platform for rapid iteration
- **Partner Strategy:** Establish relationships with key systems integrators
- **Documentation Excellence:** Invest heavily in developer and merchant documentation

Questions & Discussion

Key Discussion Points

- Resource allocation and team structure preferences
- Platform prioritization - Magento first vs. parallel development
- Partnership strategy and marketplace approach
- Success metrics and milestone definitions
- Risk tolerance and mitigation strategies