



praetorix **Foundry**

The Mobile App Launch Playbook

The complete guide to taking your app
from idea to the App Store and Google Play.

2026 EDITION

Welcome

Building a mobile application has never been easier.

AI has dramatically lowered the barrier to software development. It can help generate ideas, designs, and even working code, but building a successful product still requires thoughtful decisions at every stage.

This playbook guides you through that journey, from product discovery to launch and beyond.

Whether you're a founder, entrepreneur, product manager, or business leader, use it as a practical roadmap for building production-ready mobile applications.

Every product is different.

Use this playbook as a guide, not a strict checklist. Some sections may not apply to your product, while others may require additional research. The examples, tools, and recommendations throughout this guide are intended to help you make informed decisions, not prescribe a single way of building a product.

Let's get started.

Table of contents

Step 1: Product Discovery	4
Step 2: Market Research	9
Step 3: Requirements & Scope	14
Step 4: UX Design	17
Step 5: UI Design	22
Step 6: Backend Development	26
Step 7: Mobile App Development	31
Step 8: Infrastructure & DevOps	35
Step 9: Quality Assurance & Testing	40
Step 10: Product Analytics & Marketing	45
Step 11: Security & Data Privacy	49
Step 12: App Store & Google Play	53
Step 13: Launch & Post-Launch	57
Final Thoughts	61
About Praetorix Foundry	62

Step 1: Product Discovery

Every successful application starts with a clear idea.

Understand the problem you're solving, who you're solving it for, and why your solution matters. The clearer your vision, the easier every decision becomes.

Product Name

Choose a name that reflects your product and verify its availability.

Check:

- Domain name
- Apple App Store
- Google Play
- Trademark (if applicable)

 **Tip:** Choose a name that's easy to remember, pronounce, and search for.

 **Popular Tools:** Namecheap • GoDaddy • Cloudflare Registrar

 **Common Mistake:** Building your brand before checking availability.

Product Vision

Describe your product in one or two simple sentences.

💡 **Tip:** Everyone on the team should be able to explain it in under 30 seconds.

★ **Popular Tools:** Notion • Confluence • Google Docs

⚠️ **Common Mistake:** Confusing your product vision with a feature list.

Problem Statement

Clearly define the problem your product solves.

💡 **Tip:** If the problem isn't clear, the solution won't be either.

★ **Popular Tools:** Miro • FigJam • Notion

⚠️ **Common Mistake:** Starting with a solution before validating the problem.

Target Audience

Identify who your first customers will be.

Examples:

- Consumers (B2C)
- Businesses (B2B)
- Healthcare Providers
- Educational Institutions
- Internal Teams

 **Tip:** Start with one audience. You can always expand later.

 **Popular Tools:** ChatGPT • Claude • Perplexity

 **Common Mistake:** Trying to build an app for everyone.

User Personas

Define your primary user types and their goals.

For each persona, answer:

- Who are they?
- What are they trying to accomplish?
- What challenges do they face?

 **Tip:** Focus on primary personas, not every possible user.

 **Popular Tools:** ChatGPT • Claude • Miro • FigJam

 **Common Mistake:** Assuming all users have the same needs and behaviors.

User Roles

List every role in your application.

Examples:

- Customer
- Provider
- Administrator
- Manager

 **Tip:** Every additional role increases design, development, and testing complexity.

 **Common Mistake:** Adding new roles after development begins.

Success Criteria

Define how you'll measure a successful launch.

Examples:

- Downloads
- Active Users
- Revenue
- Bookings
- Customer Retention

 **Tip:** Focus on one or two key metrics for your first release.

 **Popular Tools:** Firebase Analytics • Google Analytics • Mixpanel • Amplitude

 **Common Mistake:** Setting goals that can't actually be measured.

MVP Scope

Define the minimum set of features required for the first public release.

 **Tip:** If a feature doesn't validate your product, it probably isn't part of your MVP.

 **Popular Tools:** Jira • Linear • Trello • Notion

 **Common Mistake:** Delaying launch by continuously adding features.

Future Roadmap

Capture ideas and features planned after the MVP.

 **Tip:** Write every idea down. You can prioritize it later.

 **Popular Tools:** Productboard • Jira • Notion • Trello

 **Common Mistake:** Prioritizing roadmap features without user feedback.

Self-Check

Before moving to the next step, make sure you can answer:

-  What problem does your product solve?
-  Who is it built for?
-  How will you measure a successful launch?
-  What is included in version 1.0?
-  What can wait until future releases?

Step 2: Market Research

A great product can still fail in the wrong market.

Market research helps you understand your customers, competitors, and opportunities before investing in development.

Target Market

Define the countries or regions where your product will be available.

Examples:

- United States
- Canada
- European Union
- United Kingdom
- Australia

 **Tip:** Choose your first market deliberately. It shapes many product decisions.

 **Popular Tools:** Statista • Similarweb • Google Trends

 **Common Mistake:** Treating every market the same.

Competitor Analysis

Identify your direct and indirect competitors.

Consider:

- Features
- Pricing
- User Experience
- Strengths
- Weaknesses

 **Tip:** Look for gaps competitors consistently fail to solve.

 **Popular Tools:** Similarweb • Sensor Tower • App Store • Google Play

 **Common Mistake:** Copying competitors without understanding why their choices

Competitive Advantage

Define why users should choose your product instead of existing alternatives.

 **Tip:** Focus on what makes switching to your product worthwhile.

 **Common Mistake:** Assuming lower pricing alone is a sustainable advantage.

Supported Languages

Determine which languages your application will support at launch.

 **Tip:** Launch with the languages your first customers actually need.

 **Popular Tools:** Lokalise • Crowdin • POEditor

 **Common Mistake:** Adding translations without planning for ongoing updates.

Regional Requirements

Identify country-specific requirements.

Examples:

- Taxes
- Payments
- Privacy Regulations
- Local Integrations
- Business Rules

 **Tip:** Validate regional requirements before development begins.

 **Common Mistake:** Assuming regulations are the same across all markets.

Pricing Strategy

Define how your product will generate revenue.

Examples:

- Free
- Subscription
- One-Time Purchase
- Freemium
- Marketplace Fees

 **Tip:** Keep your pricing simple for your first release.

 **Common Mistake:** Choosing a pricing model after development has already started.

Go-to-Market Strategy

Define how users will discover your product.

Examples:

- Organic Search
- App Store Optimization (ASO)
- Social Media
- Paid Advertising
- Partnerships
- Referrals

 **Tip:** Plan how you'll acquire your first 100 users before launch.

 **Popular Tools:** Google Ads • Meta Ads • LinkedIn Ads • Product Hunt

 **Common Mistake:** Assuming users will find your app automatically after launch.

Self-Check

Before moving to the next step, make sure you can answer:

-  Which market are you launching in?
-  Who are your main competitors?
-  What makes your product different?
-  How will your product generate revenue?
-  How will users discover it?

Step 3: Requirements & Scope

Ideas are easy. Clear requirements turn ideas into products.

Define how your application should work before design and development begin.

Functional Requirements

Define what each feature should accomplish.

 **Tip:** Describe behavior, not implementation.

 **Popular Tools:** Notion • Google Docs • Confluence

 **Common Mistake:** Writing vague or ambiguous requirements.

Non-Functional Requirements

Define how your application should perform beyond its features.

Examples:

- Performance
- Reliability
- Accessibility
- Security
- Scalability

 **Tip:** Make expectations specific and measurable.

 **Common Mistake:** Treating quality as a testing task.

Platforms

Choose which platforms will be included in the initial release.

Examples:

- iOS/Android
- What devices will be supported
- Web

 **Tip:** Choose platforms based on your users, not personal preference.

 **Common Mistake:** Trying to launch on every platform at once.

Device & OS Support

Define the minimum operating system versions and device compatibility.

 **Tip:** Balance compatibility with development effort.

 **Common Mistake:** Assuming every user has the latest device or operating system.

Out of Scope

Clearly define what is not included in the initial release.

 **Tip:** Be as clear about exclusions as inclusions.

 **Common Mistake:** Assuming everyone has the same understanding of the scope.

Self-Check

Before moving to the next step, make sure you can answer:

-  What should each feature do?
-  What quality standards should the application meet?
-  Which platforms and devices will you support?
-  What is explicitly out of scope?

Step 4: UX Design

Great products feel easy to use.

Good UX removes friction, reduces mistakes, and helps users complete tasks with confidence.

User Journeys

Map the key journeys users will complete in your application.

 **Tip:** Remove unnecessary steps whenever possible.

 **Popular Tools:** FigJam • Miro • Whimsical

 **Common Mistake:** Designing screens without considering the full journey.

Information Architecture

Organize your application's content and screen hierarchy.

 **Tip:** Group related information together.

 **Popular Tools:** Miro • FigJam

 **Common Mistake:** Organizing content around the system instead of the user.

Navigation

Choose how users will move through your application.

Examples:

- Bottom Navigation
- Tab Bar
- Sidebar
- Drawer Menu

 **Tip:** Make common actions easy to reach.

 **Common Mistake:** Using different navigation patterns throughout the app.

Onboarding Experience

Decide how new users will be introduced to your product.

 **Tip:** Help users reach their first meaningful result quickly.

 **Popular Tools:** Figma • Maze

 **Common Mistake:** Explaining every feature before users have experienced the product.

Empty States

Design screens for situations where no content exists.

Examples:

- No bookings
- No messages
- No search results

 **Tip:** Empty states should guide users toward their next action.

 **Common Mistake:** Treating empty states as an afterthought.

Loading States

Design how loading and progress are communicated.

 **Tip:** Keep users informed while they wait.

 **Common Mistake:** Showing a frozen interface while content loads.

Error States

Define how your application communicates errors.

Examples:

- Network unavailable
- Payment failed
- Invalid credentials

 **Tip:** Every error should have a clear next step.

 **Common Mistake:** Displaying technical error messages that users don't understand.

Permissions

Decide when your application should request access to device features.

Examples:

- Camera
- Photos
- Notifications
- Location
- Microphone

 **Tip:** Explain the benefit before asking for access.

 **Common Mistake:** Requesting every permission during the first app launch.

Search & Discovery (*Optional*)

Determine how users will find content within your application.

Examples:

- Search
- Filters
- Categories
- Sorting

 **Tip:** Make important content discoverable without search.

 **Common Mistake:** Treating search as the only way to find content.

Offline Experience (*Optional*)

Define how your application behaves when connectivity is limited or unavailable.

 **Tip:** Decide which features should continue working offline.

 **Common Mistake:** Losing user progress when connectivity drops.

Self-Check

Before moving to the next step, make sure you can answer:

-  Can users easily complete their primary tasks?
-  Is navigation intuitive?
-  Have you planned loading, empty, and error states?
-  Are permissions requested at the right time?
-  Will users understand what to do if something goes wrong?

Step 5: UI Design

Beautiful interfaces attract users. Consistent interfaces build trust.

Good UI creates a familiar, polished experience that feels consistent across every screen.

Brand Identity

Define your visual identity, including colors, typography, logo, and imagery.

 **Tip:** Decide on your visual style before designing individual screens.

 **Popular Tools:** Figma • Adobe Color • Coolers

 **Common Mistake:** Making visual decisions screen by screen.

Design System

Decide whether you'll build your own design system or adopt an existing one.
Examples:

- Material Design
- Atlassian Design System
- IBM Carbon
- Fluent
- Custom Design System

 **Tip:** Start with an existing design system unless your product has unique requirements.

 **Popular Tools:** Figma • Storybook • Zeroheight

 **Common Mistake:** Creating custom components before they're actually needed.

Screen Inventory

Identify every screen your application will require.

Examples:

- Splash
- Login
- Home
- Search
- Profile
- Settings
- Checkout

 **Tip:** Review every user journey to make sure no screens are missing.

 **Popular Tools:** Figma • FigJam

 **Common Mistake:** Discovering missing screens during development.

Responsive Layouts

Define how layouts should adapt to different screen sizes and orientations.

 **Tip:** Design for content, not screen dimensions.

 **Common Mistake:** Designing for only one device size.

Dark Mode (*Optional*)

Decide whether your application will support Dark Mode.

 **Tip:** Support Dark Mode only if you're prepared to maintain it.

 **Common Mistake:** Treating Dark Mode as a simple color inversion.

Icons & Illustrations

Choose a consistent icon library and illustration style.

 **Tip:** Choose icons that are clear, recognizable, and easy to understand.

 **Popular Tools:** Material Symbols • SF Symbols • Phosphor Icons • Lucide

 **Common Mistake:** Mixing multiple icon styles.

Animations & Motion

Identify where animations improve the user experience.

Examples:

- Screen Transitions
- Success States
- Micro-interactions

 **Tip:** Use motion to provide feedback and guide attention.

 **Common Mistake:** Using animation where it adds no value.

Design Review

Review every screen before development begins.

 **Tip:** Review designs from both the user and developer perspective.

 **Common Mistake:** Leaving important design decisions unresolved before development.

Interactive Prototype (*Optional*)

Create a clickable prototype covering your primary user flows.

 **Tip:** Prototype complex interactions before development.

 **Popular Tools:** Figma • ProtoPie • Framer

 **Common Mistake:** Building a detailed prototype without a clear testing goal.

Self-Check

Before moving to the next step, make sure you can answer:

-  Is the visual identity clearly defined?
-  Is the design system ready?
-  Are all required screens designed?
-  Do layouts work across supported screen sizes?

Step 6: Backend Development

The backend powers everything your users don't see.

It handles business logic, manages data, integrates with external services, and keeps your application running reliably.

Backend Strategy

Choose how your application's backend will be implemented.

Examples:

- Backend-as-a-Service (BaaS)
- Custom Backend
- Serverless

 **Tip:** Compare development time, cost, and flexibility before choosing a backend approach.

 **Popular Tools:** Firebase • Supabase • AWS Amplify • Appwrite

 **Common Mistake:** Building a custom backend before it's actually needed.

API Development

Build the APIs your application requires. Need to include API rate limiting & throttling based on expected used base estimation

Examples:

- REST
- GraphQL
- gRPC

 **Tip:** Keep your APIs consistent and easy to understand.

 **Common Mistake:** Building APIs without a versioning strategy.

Database Implementation

Design and implement the data model that supports your application.

 **Tip:** Design your database around real product workflows, not hypothetical future needs.

 **Popular Tools:** PostgreSQL • MySQL • MongoDB • Firebase Firestore

 **Common Mistake:** Changing the data model without a clear migration plan.

Authentication & Authorization

Implement secure user authentication and access control.

 **Tip:** Authentication confirms who a user is. Authorization determines what they can access.

 **Popular Tools:** Auth0 • Clerk • Firebase Authentication • Amazon Cognito

 **Common Mistake:** Building a custom authentication system without a strong reason.

Business Logic

Implement the rules and workflows that make your product work.

 **Tip:** Keep business logic separate from your user interface.

 **Common Mistake:** Placing important business logic only inside the mobile app.

File Management

Implement uploading, storing, and serving user files when required.

 **Tip:** Consider file size, access control, storage costs, and retention.

 **Popular Tools:** Amazon S3 • Cloudflare R2 • Google Cloud Storage

 **Common Mistake:** Storing large files directly in the database.

Notifications

Implement backend services for email, push notifications, and SMS.

- 💡 **Tip:** Keep notification workflows separate from your core business logic.
- ★ **Popular Tools:** Firebase Cloud Messaging • Twilio • SendGrid • Amazon SES
- ⚠️ **Common Mistake:** Hardcoding notification workflows throughout the application.

Background Jobs

Identify tasks that should run asynchronously.

Examples:

- Report Generation
- Data Synchronization
- Scheduled Notifications
- Image Processing

- 💡 **Tip:** Move long-running tasks outside the user request whenever possible.
- ⚠️ **Common Mistake:** Making users wait for operations that could run in the background.

Error Handling

Implement consistent handling of backend errors.

- 💡 **Tip:** Capture enough information to diagnose and resolve issues quickly.
- ⚠️ **Common Mistake:** Returning generic errors without recording the underlying cause.

API Documentation

Document your APIs for current and future development.

 **Tip:** Keep documentation up to date as your APIs evolve.

 **Popular Tools:** Swagger • Postman

 **Common Mistake:** Treating documentation as something to do after development.

Self-Check

Before moving to the next step, make sure you can answer:

-  Are your APIs clearly defined and documented?
-  Is your data model ready to support the application?
-  Are authentication and access rules implemented securely?
-  Is business logic centralized and maintainable?
-  Are long-running tasks and errors handled correctly?

Step 7: Mobile App Development

This is where your product comes to life.

Build a fast, reliable, and intuitive experience that works across every supported device.

Project Setup

Set up the mobile application and development environment.ə

 **Tip:** A well-structured project is easier to maintain as your app grows.

 **Common Mistake:** Starting development without establishing project standards.

Core Features

Implement the core functionality planned for your MVP.

 **Tip:** Build and validate one feature at a time instead of everything at once.

 **Common Mistake:** Developing secondary features before the product's core experience is complete.

Device Features

Integrate native device capabilities where needed.

Examples:

- Camera
- Photos
- Location
- Biometrics
- Calendar
- Contacts

 **Tip:** Implement only the device features your product actually needs.

 **Common Mistake:** Adding native integrations that don't provide clear value.

Performance Optimization

Optimize the application for speed, responsiveness, and battery efficiency.

 **Tip:** Users notice slow performance long before they notice missing features.

 **Common Mistake:** Optimizing only after the app is feature-complete.

Accessibility

Ensure the application is usable by people with different accessibility needs.

Examples:

- Dynamic Text
- VoiceOver / TalkBack
- Color Contrast
- Touch Targets

 **Tip:** Accessibility improves usability for everyone.

 **Common Mistake:** Treating accessibility as a final polishing task.

Localization (*optional*)

Implement support for multiple languages and regional formats.

 **Tip:** Remember to localize dates, currencies, and units - not just text.

 **Common Mistake:** Designing layouts that only work in English.

App Configuration

Prepare production-ready application settings.

Examples:

- App Icons
- Splash Screen
- Deep Links
- Environment Configuration

 **Tip:** Keep development, staging, and production configurations separate.

 **Common Mistake:** Hardcoding production values during development.

Release Build

Generate production-ready builds for testing and store submission.

 **Tip:** Test the exact build you'll submit to the app stores.

 **Common Mistake:** Testing debug builds instead of release builds.

Self-Check

Before moving to the next step, make sure you can answer:

-  Are all MVP features implemented?
-  Does the app work well across supported devices?
-  Have native device features been integrated correctly?
-  Is the app optimized for performance?
-  Is the application ready for QA?

Step 8: Infrastructure & DevOps

A great application deserves a reliable foundation.

Infrastructure is what keeps your app online, secure, and scalable. Planning it early helps avoid outages, deployment issues, and unnecessary operational costs.

Cloud Provider

Choose where your application will be hosted.

Examples:

- Amazon Web Services (AWS)
- Microsoft Azure
- Google Cloud Platform (GCP)
- DigitalOcean

 **Tip:** Select a cloud provider that matches your team's experience and your product's needs.

 **Popular Tools:** AWS • Azure • Google Cloud • DigitalOcean

 **Common Mistake:** Choosing infrastructure that's far more complex than your product requires.

Environments

Set up separate environments for development, testing, and production.

Examples:

- Development
- Staging
- Production

 **Tip:** Never test new features directly in production.

 **Common Mistake:** Using the production environment for development and testing.

Domain & DNS

Register your domain and configure DNS settings.

 **Estimated Cost:** Typically USD \$10–20/year

 **Tip:** Purchase your domain early to secure your brand.

 **Popular Tools:** Cloudflare • Namecheap • GoDaddy • Porkbun

 **Common Mistake:** Waiting until launch to register your domain.

SSL Certificates

Ensure all environments are secured with HTTPS.

 **Tip:** HTTPS is essential for user trust and required by most modern platforms.

 **Popular Tools:** Let's Encrypt • AWS Certificate Manager • Cloudflare

 **Common Mistake:** Forgetting to secure staging environments.

CI/CD Pipeline

Automate building, testing, and deploying your application.

 **Tip:** Automation reduces manual errors and speeds up releases.

 **Popular Tools:** GitHub Actions • Bitbucket Pipelines • GitLab CI • Codemagic

 **Common Mistake:** Relying on manual deployments.

Environment Variables & Secrets

Store sensitive credentials securely.

Examples:

- API Keys
- Database Credentials
- OAuth Secrets

 **Tip:** Keep secrets outside of your source code.

 **Popular Tools:** AWS Secrets Manager • Azure Key Vault • GitHub Secrets

 **Common Mistake:** Hardcoding secrets into the application.

Backup Strategy

Ensure databases and critical files are backed up regularly.

 **Tip:** A backup is only valuable if it can be restored successfully.

 **Common Mistake:** Assuming backups work without testing them.

Monitoring & Observability

Monitor your application's health and collect the information needed to diagnose issues.

Examples:

- Metrics
- Logs
- Alerts
- Uptime Monitoring

 **Tip:** Detect problems before your users do.

 **Popular Tools:** Datadog • Grafana • CloudWatch • UptimeRobot

 **Common Mistake:** Monitoring servers but not the application itself.

Scalability Planning

Prepare your infrastructure for future growth.

 **Tip:** Design for growth, but don't overbuild your MVP.

 **Popular Tools:** AWS Lambda • ECS • K8s • DB scaling • Redis

 **Common Mistake:** Paying for enterprise-scale infrastructure before it's needed.

Self-Check

Before moving to the next step, make sure you can answer:

-  Is your infrastructure ready for production?
-  Are development and production environments separated?
-  Is your deployment process automated?
-  Are backups, monitoring, and alerts configured?
-  Can your infrastructure support future growth?

Step 9: Quality Assurance & Testing

A successful launch depends on more than writing great code.

Testing helps ensure your application works reliably across devices, operating systems, and real-world scenarios before it reaches your users.

Test Plan

Define what should be tested before every release.

 **Tip:** A structured test plan helps ensure nothing important is overlooked.

 **Common Mistake:** Testing only new features while ignoring existing functionality.

Functional Testing

Verify that every feature works as intended.

 **Tip:** Test complete features—not just individual screens.

 **Common Mistake:** Assuming a feature works because one scenario was successful.

User Journey Testing

Test complete user journeys from beginning to end.

Examples:

- Registration
- Login
- Booking
- Checkout
- Subscription

 **Tip:** Most critical issues appear between screens, not on them.

 **Common Mistake:** Testing individual pages instead of complete workflows.

Cross-Platform Testing

Verify your app on every supported platform.

Examples:

- iOS
- Android
- Tablets

 **Tip:** The same feature may behave differently across platforms.

 **Common Mistake:** Assuming iOS and Android behave identically.

Device Testing

Test your application on different screen sizes and physical devices.

 **Tip:** Simulators are helpful—but they don't replace real devices.

 **Common Mistake:** Releasing an app that has only been tested on one phone.

Performance Testing

Measure responsiveness, loading times, and overall application performance.

 **Tip:** Performance is part of the user experience.

 **Common Mistake:** Optimizing performance only after launch.

Network Testing

Verify how the app behaves under different network conditions.

Define Offline-first capabilities / sync strategy if required

Examples:

- Slow connection
- Lost connection
- Airplane Mode

 **Tip:** Users won't always have a perfect internet connection.

 **Common Mistake:** Only testing on fast Wi-Fi.

Edge Case Testing

Test unusual and unexpected scenarios.

Examples:

- Empty data
- Invalid input
- Expired sessions
- Duplicate actions

 **Tip:** Edge cases are where many production bugs are discovered.

 **Common Mistake:** Testing only the "happy path."

Beta Testing

Release your app to a limited group of users before launch.

Examples:

- TestFlight
- Google Play Internal Testing

 **Tip:** Real users often discover issues your team never considered.

 **Common Mistake:** Skipping beta testing because "everything works on our devices."

Bug Review

Review, prioritize, and resolve reported issues before release.

 **Tip:** Not every bug is equally important—but every critical issue should be addressed.

 **Common Mistake:** Launching with unresolved critical bugs because deadlines are approaching.

Self-Check

Before moving to the next step, make sure you can answer:

-  Have all core features been tested?
-  Has the app been tested on real devices?
-  Have complete user journeys been verified?
-  Have poor network conditions been considered?
-  Has the application been tested by real users?

Step 10: Product Analytics & Marketing

Product analytics and marketing attribution help you understand how users discover your product, how they interact with it, and where they drop off.

Success Metrics (KPIs)

Define how you'll measure the success of your product.

Examples:

- Daily Active Users (DAU)
- Monthly Active Users (MAU)
- Conversion Rate
- Customer Retention
- Revenue

 **Tip:** Focus on a handful of meaningful metrics instead of tracking everything.

 **Popular Tools:** Firebase Analytics • Google Analytics 4 • Mixpanel • Amplitude

 **Common Mistake:** Tracking dozens of metrics but never using them to make decisions.

Event Tracking

Identify the key actions users perform inside your application.

Examples:

- Sign Up
- Login
- Booking Created
- Purchase Completed
- Subscription Started

 **Tip:** Every important feature should generate meaningful analytics events.

 **Popular Tools:** Firebase Analytics • Mixpanel • Amplitude

 **Common Mistake:** Launching without verifying that events are actually being recorded.

Conversion Funnels

Map the key journeys you want users to complete.

Examples:

- Install → Register → Subscribe
- Browse → Purchase
- Landing Page → Download → Account Creation

 **Tip:** Funnels quickly reveal where users abandon your product.

 **Common Mistake:** Measuring downloads without understanding what users do afterward.

Marketing Attribution

Determine how you'll measure where users come from.

Examples:

- Google Ads
- Meta Ads
- TikTok Ads
- Organic Search
- Referral Campaigns

 **Tip:** Attribution should be configured before your first marketing campaign.

 **Popular Tools:** AppsFlyer • Adjust • Firebase Analytics

 **Common Mistake:** Starting paid advertising before attribution is working correctly.

Dashboards & Reporting

Create dashboards to visualize your most important product and marketing metrics.

 **Tip:** Make key metrics easy to understand for everyone on the team.

 **Popular Tools:** Looker Studio • Mixpanel

 **Common Mistake:** Collecting data without reviewing it regularly.

Self-Check

Before moving to the next step, make sure you can answer:

- ✓ Do you know how success will be measured?
- ✓ Can you track key user actions?
- ✓ Can you identify where users abandon key journeys?
- ✓ Can you measure where new users come from?
- ✓ Do you have dashboards for your key product metrics?

Step 11: Security & Data Privacy

Users trust you with their personal information.

Protecting that trust means thinking beyond passwords and encryption. Every application should be designed with security, privacy, and compliance in mind from the very beginning.

Privacy Policy

Prepare a Privacy Policy explaining how user data is collected, stored, and processed.

 **Tip:** Most mobile applications require a publicly accessible Privacy Policy before launch.

 **Popular Tools:** Termly • iubenda • TermsFeed

 **Common Mistake:** Copying another company's privacy policy without reviewing your own data practices.

Terms of Service

Prepare Terms of Service describing how users may use your application.

 **Tip:** Clear terms help protect both your business and your users.

 **Popular Tools:** Termly • TermsFeed • iubenda

 **Common Mistake:** Treating Terms of Service as an afterthought.

Consent Management

Determine when users should be asked for consent.

Examples:

- Analytics
- Marketing
- Notifications
- Location Services

 **Tip:** Request consent only when required and make it easy for users to understand and manage their choices.

 **Common Mistake:** Requesting permissions before users understand their purpose.

Data Collection

Review every type of information your application collects.

Examples:

- Personal Information
- Payment Data
- Health Data
- Location
- Device Information

 **Tip:** Collect only the information your product genuinely needs.

 **Common Mistake:** Collecting data simply because it might be useful one day.

Regulatory Compliance

Identify which regulations apply to your application.

Examples:

- GDPR
- HIPAA
- FERPA
- COPPA
- PCI DSS

 **Tip:** Compliance requirements depend on your users, your industry, and where your product operates.

 **Common Mistake:** Assuming regulations only matter after your product becomes successful.

Account & Data Deletion

Allow users to delete their accounts and personal information where required.

 **Tip:** Giving users control over their data builds trust and may be legally required.

 **Common Mistake:** Making account deletion significantly harder than account creation.

Security Review

Perform a final review before launching your application.

Consider:

- Authentication
- User Data
- Access Control
- Third-Party Services
- Production Configuration

 **Tip:** Small security improvements before launch can prevent major incidents later.

 **Common Mistake:** Assuming security is a one-time task instead of an ongoing process.

Self-Check

Before moving to the next step, make sure you can answer:

- ☒ Do users know how their data is collected and used?
- ☒ Have all required legal documents been prepared?
- ☒ Are applicable regulations understood?
- ☒ Are users able to manage or delete their personal data?
- ☒ Is your product ready to protect user data from day one?

Step 12: App Store & Google Play

Publishing your app is more than uploading a build.

Both Apple and Google have technical, legal, and content requirements that every application must meet before it becomes available to users.

Developer Accounts

Create and verify your developer accounts.

Estimated Cost:

- Apple Developer Program: USD \$99 per year
- Google Play Developer: USD \$25 one-time registration fee

 **Tip:** Set up your developer accounts early, as verification may take several days.

 **Popular Tools:** Apple Developer • Google Play Console

 **Common Mistake:** Waiting until development is complete to create developer accounts.

App Store Listing

Prepare your public store listing.

Include:

- App Name
- Subtitle
- Description
- Keywords
- Promotional Text (*if applicable*)
- Previews and Screenshots

 **Tip:** Your store page is often the first impression users have of your product.

 **Common Mistake:** Spending months building the app but only minutes writing the store description.

App Assets

Prepare all required marketing assets.

Examples:

- App Icon
- Screenshots
- Feature Graphic (Google Play)
- App Preview Video (Optional)

 **Tip:** Show how your app solves a problem, not just what it looks like.

 **Popular Tools:** Figma • Canva

 **Common Mistake:** Using screenshots that don't represent the latest version of the app.

Pricing & Availability

Choose your pricing model and the countries where your app will be available.

 **Tip:** Your launch markets should match the countries identified during the discovery phase.

 **Common Mistake:** Making the app available globally before you're ready to support those users.

In-App Purchases & Subscriptions *(If Applicable)*

Configure digital products before submitting your application.

 **Tip:** Mobile platforms have specific rules governing digital purchases and subscriptions.

 **Common Mistake:** Leaving payment configuration until the final days before launch.

Store Review

Submit your application for review and be prepared to answer reviewer questions.

 **Tip:** Review times vary and additional information may be requested before approval.

 **Common Mistake:** Planning your marketing launch before your app has actually been approved.

Release Strategy

Choose how your application will be released.

Examples:

- Manual Release
- Automatic Release
- Staged Rollout

 **Tip:** A gradual rollout allows you to detect issues before they affect all users.

 **Common Mistake:** Releasing a major update to 100% of users immediately.

Self-Check

Before moving to the next step, make sure you can answer:

- ☒ Are your developer accounts fully configured?
- ☒ Is your App Store listing complete?
- ☒ Have all required assets been prepared?
- ☒ Has the release build been tested?
- ☒ Is your launch strategy defined?

Step 13: Launch & Post-Launch

Launching your app is just the beginning.

The first few weeks after release are critical for monitoring performance, supporting users, fixing issues, and learning how your product performs in the real world.

Launch Readiness

Perform a final review before announcing your application.

Verify:

- Production Environment
- APIs
- Payments
- Analytics
- Notifications

 **Tip:** A simple pre-launch checklist can prevent major launch-day issues.

 **Common Mistake:** Launching immediately after store approval without verifying your production environment.

Marketing Launch

Coordinate your launch across all planned marketing channels.

Examples:

- Website
- Social Media
- Email Campaigns
- Product Hunt
- Press Release

 **Tip:** Even the best app won't succeed if nobody knows it exists.

 **Popular Tools:** Mailchimp • Buffer • Hootsuite • Product Hunt

 **Common Mistake:** Releasing your app without a launch plan.

Collect User Feedback

Listen to your early users and identify opportunities for improvement.

Examples:

- App Store Reviews
- Customer Support
- Surveys
- User Interviews

 **Tip:** Your first users often provide your most valuable product feedback.

 **Popular Tools:** Typeform • Google Forms • Intercom • Zendesk

 **Common Mistake:** Ignoring user feedback because it doesn't match your original assumptions.

Customer Support

Provide users with a simple way to ask questions or report problems.

 **Tip:** Fast, helpful support builds trust and improves retention.

 **Popular Tools:** Intercom • Zendesk • Freshdesk • Crisp

 **Common Mistake:** Launching without a defined support process.

Updates & Bug Fixes

Plan regular updates to improve your application after launch.

 **Tip:** Version 1.0 is the beginning of your product, not the end.

 **Common Mistake:** Waiting too long to fix issues reported by users.

Plan the Next Release

Prioritize improvements based on user feedback and product analytics.

 **Tip:** The best products evolve continuously after launch.

 **Popular Tools:** Jira • Linear • Productboard • Notion

 **Common Mistake:** Trying to implement every feature request instead of focusing on the highest impact improvements.

Self-Check

Before considering your launch complete, make sure you can answer:

- ✓ Is your application running reliably in production?
- ✓ Are users actively using your product?
- ✓ Are you monitoring performance and collecting feedback?
- ✓ Do users know how to contact your team?
- ✓ Do you have a clear plan for your next release?

Final Thoughts

Launching your application isn't the finish line.

It's the moment your product meets real users for the first time.

From that point forward, every release, every piece of feedback, every metric, and every customer interaction becomes an opportunity to improve.

The most successful products aren't the ones that launch perfectly. They're the ones that continue to learn, adapt, and evolve.

We hope this playbook has given you a clearer understanding of what it takes to build and launch a production-ready mobile application.

Remember:

- Every successful product is built through thoughtful decisions.
- Ship, learn, improve, and repeat.
- Progress beats perfection.

Good luck with your launch!

About Praetorix Foundry

Praetorix Foundry helps startups and businesses turn ideas into production-ready digital products.

From product strategy and UX/UI design to mobile development, cloud infrastructure, app store publishing, and post-launch support, we help teams bring ambitious ideas to life.



foundry.praetorix.com