

Request for Proposal: Custom Software Development

About this template: Use this RFP to hire a software development company, such as an agency or development partner, to design, build, launch and support a custom application. It isn't for buying software that already exists. Fill in the [square brackets], and replace the example entries, which describe a booking and field service app, with your own. Delete the italic instructions before you send the RFP, and have legal counsel review Sections 9 and 15 and your draft contract first.

Item	Details
RFP title	[Project name]: custom software development
RFP number	[RFP-YYYY-NNN]
Issued by	[Organization legal name]
Issue date	[Date]
Questions due	[Date, time and time zone]
Proposals due	[Date, time and time zone]
RFP contact	[Name, title, email]

1. Introduction and background

1.1 About us

[Two or three sentences: who you are, what you do, where you operate and how large you are.]

1.2 Background and the problem

[Describe the problem the software will solve: how the work is done today, which systems are involved, who is affected and why you're acting now.]

Example: Customers book service visits by phone and can't see when a technician will arrive, so the call center spends much of its day answering status questions.

Technicians receive job details by email.

1.3 Purpose of this RFP

[Organization name] ("we", "us") invites proposals from software development companies to design, build, test and launch [short description of the application],

and to support it after launch. We intend to select one development partner (the "vendor") through the process in Section 14. This RFP is not an offer and does not commit us to award a contract.

1.4 Attachments

Attachment	Description
A	Requirements workbook: the tables in Sections 4, 5 and 10, for vendors to complete
B	Pricing workbook: the tables in Section 12
C	[Wireframes, designs or a system diagram, if you have them]
D	[Draft contract, such as a master services agreement and statement of work]
E	[Security questionnaire, if you use one]

2. Objectives

List the outcomes you want, in priority order. Describe results rather than features, and say how you'll know you've achieved each one.

No.	Objective	How we will measure it
O1	[Example: Let customers book and reschedule service visits online, without calling us]	[Example: Share of bookings made online, [target]% by [date]]
O2	[Example: Show customers their technician's arrival window, updated during the day]	[Example: Status calls to the call center per week]
O3	[Example: Launch the first release before [date or event]]	[Example: Apps live in the app stores and the web application in production]
O4	[Objective]	[Measure]

3. Scope

Give vendors the figures they'll size and price against. A vendor that has to guess will either add contingency or leave work out.

3.1 Users

User type	What they do in the application	Number of users	Platforms
[Example: Customers]	[Book, reschedule and track visits; pay]	[At launch, and expected in [two]]	[Web, iOS, Android]

User type	What they do in the application	Number of users	Platforms
	invoices]	years]	
[Example: Technicians]	[See daily jobs, update job status, capture photos and signatures]	[Number]	[iOS / Android phones]
[Example: Dispatchers]	[Assign jobs and manage schedules]	[Number]	[Web]
[Example: Administrators]	[Manage users, settings and content]	[Number]	[Web]

3.2 Platforms

Platform	In scope	Details
Web application	[Yes / No]	[Responsive for desktop and mobile browsers; supported browsers]
iOS app	[Yes / No]	[Minimum iOS version; iPhone, iPad or both]
Android app	[Yes / No]	[Minimum Android version; phones, tablets or both]
Administration portal	[Yes / No]	[Web, for our staff only]
API for partners	[Yes / No]	[Who will use it, and for what]

3.3 Integrations

System	Purpose	Data exchanged and direction	Interface available	Owner
[Example: ERP]	[Customer accounts and invoices]	[Accounts and invoices to the app; payments back to the ERP]	[REST API / file export / none / unknown]	[Team or supplier]
[Identity provider]	[Staff sign-in]	[Single sign-on]	[SAML 2.0 / OpenID Connect]	[IT]
[Payment provider]	[Card payments]	[Payment requests and results]	[API]	[Finance]
[Email and SMS service]	[Notifications]	[Outbound messages]	[API]	[IT]

3.4 Data

- **Records and volumes.** [Main records, e.g. customers, visits, jobs, photos, invoices; numbers at launch, growth per year and peak transactions per hour.]

- **Personal and sensitive data.** [Types of personal data, and any regulated data such as payment, health or financial information.]
- **Migration.** [None / Migrate [records] covering [number] years from [current system]. State who extracts and cleans the data.]
- **Location and retention.** [Regions where data must be stored and processed; how long records are kept.]

3.5 In scope

1. **Discovery:** [a prioritized backlog, user journeys, architecture and a refined estimate].
2. **Design:** [UX and UI design for each platform, following our brand guidelines].
3. **Development:** [the applications, APIs and integrations in Sections 3 and 4].
4. **Testing:** [as in Section 7, including support for our user acceptance testing].
5. **Infrastructure:** [environments, CI/CD and infrastructure as code in our cloud account].
6. **Launch:** [app store submission under our accounts, and production deployment].
7. **Handover:** [documentation, runbooks and knowledge transfer].
8. **Warranty and support:** [as in Section 11].

3.6 Out of scope

- [Example: Hosting fees, which we pay directly to the cloud provider]
- [Example: Changes to the ERP itself]
- [Example: Content writing and translation]

3.7 Phases

Splitting the work into phases lets you fix the price of what you can define now and keep options open for the rest.

Phase	What it covers	Pricing requested
1: Discovery	[Requirements, backlog, key designs, architecture, refined estimate]	Fixed price (Section 12.1)
2: First release	[All Must-have requirements in Sections 4, 5 and 10]	[Fixed price (12.2) / time and materials with a not-to-exceed amount (12.3) / both]

Phase	What it covers	Pricing requested
3: Later releases	[Nice-to-have requirements and future items]	[Rate card in 12.3 and an indicative estimate]
Support	Warranty, hypercare and optional ongoing support	Section 12.5

We may [award Phase 1 only and decide on Phase 2 after reviewing the discovery outputs / award Phases 1 and 2 together]. Our target start date is [date], and we need the first release live by [date, and the reason].

4. Functional requirements

Write each requirement as a user story (who needs to do what, and why) or as a single requirement line, one need per line. Mark each one Must-have (in the first release and in the price) or Nice-to-have (priced as an option or planned for later). Keep the Must-have list short. Replace the example rows with your own.

4.1 Response codes

Give each requirement in Sections 4, 5 and 10 one of these codes, and explain it in the comment column.

Code	What it means
Included	Built as part of the priced scope
Vendor component	Delivered using code, a framework or a tool the vendor already owns (see Section 9.2)
Third party	Delivered using a third-party service or open-source component; name it, its license and any cost
Optional	Can be delivered but isn't in the base price; price it in Section 12
Clarify	Can't be estimated without more information; state your question or assumption
Not proposed	Not included; explain why

Product RFPs use codes such as standard, configuration and roadmap, because they ask about a product the vendor already sells. These codes show instead how each requirement will be built, who owns that part and whether it's in the price.

4.2 Requirements

ID	User story or requirement	Priority	Response code	Vendor comment
FR-01	[Example: As a customer, I want to create an account with my email address [or sign in with Apple or Google], so that I can manage my bookings.]	[Must-have]		
FR-02	[Example: As a customer, I want to book a visit by choosing a service, my address and an available time slot, so that I don't have to call.]	[Must-have]		
FR-03	[Example: As a customer, I want to reschedule or cancel a visit up to [24] hours before it starts, so that I can fit it around my plans.]	[Must-have]		
FR-04	[Example: As a technician, I want to see my jobs for the day and update each job's status, including when I have no signal, so that dispatch knows where I am.]	[Must-have]		
FR-05	[Example: As a dispatcher, I want to assign and reassign jobs on a calendar by technician, so that I can respond to changes during the day.]	[Must-have]		

ID	User story or requirement	Priority	Response code	Vendor comment
FR-06	[Example: As an administrator, I want to manage users and roles without a developer, so that we can onboard staff ourselves.]	[Must-have]		
FR-07	[Example: As a customer, I want to pay an invoice by card in the app, so that I don't have to phone with my card details.]	[Nice-to-have]		
FR-08	[Requirement]	[Priority]		

4.3 Acceptance criteria

Add acceptance criteria to the stories that carry the most cost or risk. They tell vendors what "done" means, and they become the tests you'll use to accept the work.

Example for FR-03:

- Given a visit that starts more than [24] hours from now, when the customer chooses a new available slot, then the visit moves to that slot, the old slot is released and the customer receives a confirmation.
- Given a visit that starts within [24] hours, when the customer opens it, then the reschedule and cancel options are hidden and the app shows our phone number.
- Every change appears in the visit's history with the date, time and who made it.

5. Non-functional requirements

The bracketed values are placeholders, not recommendations. Set targets that fit your users and budget, because each one affects the price.

ID	Requirement	Priority	Response code	Vendor comment
NF-01	Performance: [screens load / API calls respond] within [time] for [percentage]% of requests at [number] concurrent users, and capacity for [growth figures] within [two] years. Describe how you'll test this before launch.	[Must-have]		
NF-02	Availability: [target]% in production, measured monthly, excluding maintenance announced [notice period] in advance. Describe what your design and the hosting need to meet it.	[Must-have]		
NF-03	Backup and recovery: a recovery point objective (RPO) of [time] and a recovery time objective (RTO) of [time], with a restore test before launch.	[Must-have]		
NF-04	Accessibility: the web application and mobile apps conform to WCAG [2.1 / 2.2] Level [A / AA]. Describe how you'll test and what evidence you'll deliver.	[Must-have]		

ID	Requirement	Priority	Response code	Vendor comment
NF-05	Devices and browsers: [browsers and versions], iOS [version] and later, Android [version] and later.	[Must-have]		
NF-06	Offline use: [the technician app works without a connection for [tasks] and syncs when the connection returns].	[Priority]		
NF-07	Monitoring: application logs, error tracking, uptime monitoring and alerts to [team or tool], in place before launch.	[Must-have]		
NF-08	Maintainability: code follows documented standards, has automated tests for [agreed critical paths], and can be built and deployed by another competent team using the documentation and pipelines you deliver.	[Must-have]		
NF-09	Privacy: the application collects only the personal data its features need, and supports our obligations under [laws that apply], including access and deletion requests.	[Must-have]		

Public bodies: check whether a law or policy sets the accessibility standard you must meet, and use that version and level.

6. Technical constraints

Separate firm constraints from preferences. A preferred stack helps if your own team will maintain the code, but don't rule out a better proposal by prescribing technology you don't need to.

Area	Our standard or preference	Firm or preference
Hosting	[Our [cloud provider] account / vendor-hosted / on-premises]	[Firm / Preference]
Hosting region	[Where production data must be hosted]	[Firm / Preference]
Languages and frameworks	[Preferred stack, if any]	[Firm / Preference]
Database	[Preferred database, if any]	[Firm / Preference]
Mobile approach	[Native / cross-platform / no preference]	[Firm / Preference]
Identity	[Identity provider for staff; approach for customer accounts]	[Firm / Preference]
Source control and CI/CD	[Our organization in [code hosting platform]]	[Firm / Preference]

In your proposal:

- Recommend an architecture and technology stack, and explain why it suits our requirements and our team. Where you depart from a preference above, explain the benefit and the effect on maintaining the code.
- If we have no mobile preference, recommend native or cross-platform development and explain the trade-offs.
- Describe the environments you'll set up (for example development, test, staging and production) and who pays for each.
- List any proprietary platform, low-code product or paid tool your approach depends on, with its license terms and its cost to us after the contract ends.

7. Delivery approach

Answer each question in no more than [300] words.

- **D1. Method.** Describe your delivery method (for example Scrum or Kanban) and how you'd run it for this project. How many hours a week do you need from our product owner?
- **D2. Discovery.** What does your discovery phase produce, how long does it take, and who takes part?
- **D3. Sprints and demos.** State your sprint length, and confirm that you'll demonstrate working software at the end of every sprint in an environment our team can use.
- **D4. Estimates and changes.** How do you estimate and track progress, and how early will you warn us if the timeline or budget is at risk? How do you handle changed requirements under a fixed price and under time and materials?
- **D5. Definition of done.** Provide your definition of done. It must include at least the items listed after these questions.
- **D6. Testing.** Describe your unit, integration and end-to-end testing; performance, security and accessibility testing; testing on real devices; and how you'll support our user acceptance testing.
- **D7. CI/CD and releases.** Describe your pipeline: automated builds and tests on every change, code review, deployment to each environment, infrastructure as code, release approval and rollback.
- **D8. Reporting.** What will you report, and how often? Include progress against the backlog, budget used against the estimate, risks and decisions you need from us.
- **D9. Launch and handover.** Describe your launch plan, including app store submission, and the documentation and knowledge transfer you'll provide so that our team, or another supplier, can run and change the software without you.

Our minimum definition of done. A user story is done only when:

- the code has been reviewed by a second developer and merged through the pipeline;
- its automated tests pass, and no existing tests fail;
- it meets its acceptance criteria and has been demonstrated to our product owner;

- it is deployed to the [staging] environment, with accessibility and security checks passed;
- affected documentation is updated; and
- no [Severity 1 or 2] defects are open against it.

8. Team and staffing

Ask for the people who will do the work. Vendors can add rows.

Role	Name	Location (city, country)	Employee or subcontractor	Time on project (%)
Delivery or project manager				
Business analyst				
Technical lead or architect				
UX and UI designer				
Developers (one row each)				
QA or test engineers				
DevOps or cloud engineer				

Attach a short résumé for each named person, covering similar projects and the technologies you propose.

- **Key personnel.** The [delivery manager, technical lead and lead designer] may not be replaced without our written approval, except for reasons outside your control, and any replacement must have equivalent experience.
- **Working hours.** At least [number] hours of overlap each working day with [time zone].
- **Location.** State where each person will work. [On-site attendance is required for [workshops / launch / none].]
- **Subcontracting.** Name every subcontractor and freelancer, their work and their location. Don't subcontract beyond what your proposal names without our written approval. You remain responsible for subcontractors' work.

- **Background checks.** [Required, where local law allows, for anyone with access to production systems or personal data.]

9. Intellectual property and source code

Agree your position with legal counsel before you issue the RFP, and put the final terms in the contract. Who owns software depends on what the contract says and on the law that applies, so don't assume that paying for development makes you the owner. This template isn't legal advice.

Confirm that you accept each position below, or list your exceptions in Section 16.

9.1 Custom work

[Option A: We will own the source code, designs, documentation and other materials created for us under the contract (the "custom work") from [creation / payment], and you will sign any documents needed to confirm that.]

[Option B: You will own the custom work and grant us a [perpetual, irrevocable, worldwide, royalty-free] license to use, copy, modify and maintain it, and to have other suppliers do so for us. [State whether the license is exclusive, and whether you may reuse the custom work for other clients.]]

Keep one option. Option A suits software that's core to your business, or that your own team or another supplier will maintain. A vendor may price Option B lower if it can reuse the work.

9.2 Pre-existing materials

List any code, frameworks or tools you already own and plan to include in the software ("pre-existing materials"). You keep ownership of them, and grant us a [perpetual, irrevocable, royalty-free, non-exclusive] license to use, modify and maintain them as part of the software, and to have other suppliers do so for us, including after the contract ends. [Deliver them as source code.]

9.3 Source code and accounts

- All source code, build scripts, infrastructure-as-code files and automated tests are committed to our repository from the first sprint, with full history, and we can build and deploy the software using your documentation.
- Cloud, app store, domain and third-party service accounts are held in our name. Credentials and signing keys are handed over to us.