Nepal Digest

LIFE. CULTURE. PEOPLE & NEPAL



AI & TECHNOLOGY / APPS & SOFTWARE

Code made in Kathmandu

Nepal’s software builders are finding global customers without leaving home.

AUGUST 18, 2026

Code made in Kathmandu

Nepal’s software builders are finding global customers without leaving home.

By Nepal Digest - August 18, 2026 - Nepal

Code made in Kathmandu can serve a customer far beyond Nepal, but successful software work is not simply a matter of writing a program and sending it abroad. It depends on understanding a customer, defining the work, protecting information and supporting what has been delivered.

This analysis looks at the practical capabilities behind a software-export business. It uses dated institutional sources to establish context and distinguishes the sector’s opportunity from a promise of easy earnings or guaranteed work for any individual developer.

> Code delivery checklist: define the customer problem, agree on requirements
> Guide illustration: Nepal Digest.

Table of Contents

- Code is a service before it becomes an export statistic
- Start with a problem a customer can recognise
- Agree on what the code must do
- Communication is part of the product
- Protect data through ordinary working habits
- Build a business that survives beyond one developer
- Judge the opportunity by repeatable delivery

Code is a service before it becomes an export statistic

For a client, buying software means asking another team to solve a problem reliably. The work can include understanding a process, designing an interface, connecting systems, migrating data and maintaining the result. The code is essential, but it is one part of a longer relationship.

IIDS’s June 2023 IT-industry workshop brought researchers and companies together to discuss the sector’s development and constraints. Its published account identifies issues including skills, infrastructure, payment systems and institutional support. That is a documented discussion of industry conditions, rather than a current ranking of individual firms.

Kathmandu’s opportunity is therefore not explained by a laptop and an internet connection alone. A team also needs a way to find work, agree on requirements, communicate progress and remain accountable after delivery. These less visible capabilities determine whether a first assignment becomes a continuing business relationship.

Start with a problem a customer can recognise

A prospective customer should be able to understand what will improve. A booking system might reduce repeated manual work; a reporting tool might make information available sooner. Describing the problem clearly is more useful than listing every programming language a team can use.

A small demonstration can establish credibility if it is honest about its limits. Show what works, what remains unfinished and which assumptions were used. A portfolio should distinguish a personal project from paid client work and avoid implying that a public demo has been tested under conditions it has never encountered.

Choose a first assignment that the team can support. An impressive promise can create a difficult obligation when requirements, time zones and maintenance needs become real. A narrower project delivered dependably often creates better evidence of competence than a sweeping proposal completed only

in part.

Agree on what the code must do

Misunderstandings often begin with familiar words that mean different things to the buyer and developer. A request for a dashboard, a secure login or an automated report can conceal several separate requirements. Written examples help turn those words into behaviour that both sides can recognise.

Before work begins, clarify the intended users, required features, delivery stages, review process and responsibility for supplying information. Agree on how changes will be requested. A conversation that produces a new feature should also address its effect on the schedule and cost rather than quietly expanding the original commitment.

Acceptance needs to be understandable. The customer should know how to check the result, and the team should know what constitutes completion. A handover that depends on an unwritten explanation from one developer creates avoidable uncertainty for everyone who must operate the software later.

Communication is part of the product

Cross-border work can be delayed by uncertainty even when the technical work is progressing. A customer may not know whether a question is awaiting an answer, whether a demonstration is

ready or whether a problem has changed the plan. Short, dependable updates can prevent that uncertainty from becoming distrust.

Choose a routine suited to the project. Record decisions after meetings, identify the next action and name the person responsible. When a delay becomes likely, explain it early enough for the client to make a decision. An optimistic message that hides a known obstacle is usually less useful than an accurate revised plan.

Time-zone differences also need an agreement. Not every question should require someone in Nepal to remain available throughout the night. Define overlapping hours, expected response times and the process for genuine incidents. A service that depends on permanent exhaustion is difficult to sustain as the customer base grows.

Protect data through ordinary working habits

Teams may receive information that a client would never publish openly. Access should be limited to the work that requires it, with appropriate account controls and an agreed process for removing access when someone leaves the project. Convenience should not turn a temporary task into permanent access for everybody.

Use realistic but non-sensitive examples where possible in demonstrations and training. Do not place client credentials, personal data or private business records into public repositories, shared screenshots or unapproved external tools. Agree on which services can be used before sending project material to them.

Security is also a maintenance responsibility. Document dependencies, update procedures, backups and whom to contact if a problem is discovered. The relevant controls depend on the system and its risks. A statement that the code is secure should be supported by a defined review process rather than used as a general marketing phrase.

Build a business that survives beyond one developer

A company can appear to scale while remaining dependent on one person who knows every deployment, customer and exception. Documentation, shared understanding and a clear review process reduce that fragility. They also make it easier for junior staff to contribute without guessing how the system works.

Training should include the judgement around code: reading an existing project, asking a useful question, explaining a trade-off and recognising when a change may affect another part of the system. These abilities take practice on real work with appropriate supervision. They cannot be replaced by a list of course

certificates.

Commercial administration matters alongside engineering. Businesses need appropriate advice on contracts, taxation, payment arrangements and applicable rules. This article does not prescribe a universal legal or tax structure. It identifies the administrative work that should be resolved for the actual company, client and jurisdiction.

Judge the opportunity by repeatable delivery

The World Bank’s 2025 Nepal Country Economic Memorandum examines constraints on growth and opportunities to improve exports and firm development. A national opportunity, however, does not guarantee that a particular team will find customers or that every worker will share equally in the benefits.

For a software business, useful measures include completed projects, retained customers, support obligations, staff development and dependable income. Explain what a metric covers before comparing companies. A download count or an impressive client logo may reveal less than whether a service keeps working after launch.

Code made in Kathmandu can travel widely when the work around it is dependable. That means building trust through a clear

problem, careful delivery and continuing support. Read Nepal Digest’s related articles on technology exports, global careers from Nepal and the development of F1Soft.

ABOUT THIS STORY

How we reported this

Desk research using the public institutional sources listed here. This overview does not claim original interviews or first-hand field reporting.

AI-use disclosure

Generative AI assisted the first structured draft. An editor is responsible for source selection, factual review, framing and publication.

Originally published by Nepal Digest - https://nepaldigest.com/code-made-in-kathmandu/