

# Developing an Indian DSL (Programming Language) for Accounts, Commerce, Finance, And Fintech Professionals

**Mr. Mohammed Maruff**

*PhD Scholar*

*Department of Computer Applications*

*School of Computer Information and Mathematical Sciences*

*B.S. Abdur Rahman Crescent Institute of Science and Technology,*

*Vandalur, Chennai – 600048, Tamilnadu.*

*mohammedmaruff\_cse\_july2024@crescent.education*

**Dr. S.P.Valli**

*Associate Professor*

*Department of Computer Science and Engineering*

*School of Computer Information and Mathematical Sciences*

*B.S. Abdur Rahman Crescent Institute of Science and Technology,*

*Vandalur, Chennai – 600048, Tamilnadu.*

[vallisp@crescent.education](mailto:vallisp@crescent.education)

## Abstract

This paper proposes eTamil, a novel domain-specific language (DSL) tailored for India's financial sector. It addresses the limitations of general-purpose languages by embedding Indian tax and banking regulations. The language features bilingual keywords in Tamil and Latin transliteration, aiming to democratize fintech development. Built with Rust and LLVM, eTamil offers a secure, high-performance solution for automation and compliance.

**Keywords:** *Domain-Specific Language, eTamil, Fintech, India, Tamil, GST, ITR, RBI, UPI, ULI, Rust, LLVM, Blockchain*

## Introduction

### *Background and Rationale*

With initiatives like Digital India, UPI (Unified Payments Interface) and GST (Goods and Services Tax), the economy of India is rapidly turning digital. Due to this move, companies require software capable of supporting the complicated financial regulations, tax laws and banking laws in India. General-purpose programming languages (GPLs) struggle with these requirements, and a Domain-Specific Language (DSL) may close the gap. Although the West has already adopted DSLs in its finance, India requires one that will facilitate local compliance and regional languages to ensure that the digital economy is inclusive.

## *Problem Statement*

GPLs such as Python and Java are flexible, however, it needs a great deal of specialized code to do Indian financial work, including GST reconciliation, ITR (Income Tax Return) calculation and RBI (Reserve Bank of India) compliance. They also do not have in-built support of Indian languages, thus limiting their use. Thus, one will require a new DSL, which will incorporate Indian regulatory guidelines and provide regional access to languages.

## *Objectives and Research Questions*

This review will explore the need of an Indian DSL, named it as eTamil. Will examine some existing DSLs, learn about multilingual programming, and learn about some contemporary compiler technology like Rust and LLVM. Also will look at the way in which blockchain architectures such as Hyperledger can be incorporated, too. The study will answer:

- What are the shortcomings of existing programming languages to support Indian financial systems?
- What can be done to localize a programming language?
- What regulatory and technical requirements must a financial DSL meet in India?
- What can be done to implement this DSL with Rust and LLVM?
- What can the role of the integration of blockchain be?

## *Methodology and Structure*

A narrative literature review will be utilized, in the form of scholarly journals, governmental publications and industry standards, found on the RBI and GST Council. The sources are categorized as five sections:

- DSL theory,
- Indian finance,
- multilingual programming,
- compiler design and
- blockchain in finance.

The paper is structured in this manner where each topic has its own section with a conclusion and finally the suggested role of eTamil.

This review of literature checked against 28 industrial and academic sources from 2003–2025, which were found using Google Scholar, IEEE Xplore, and official regulatory archives. Relevance to DSL development, financial technology architecture, Indian regulatory architecture, and localizing language were the selection criteria. Thematic comparative analysis was done to analyze domain fit, regulatory extent, and technical achievability.

## Motivation And Context

### *The Gap in Existing Programming Languages*

Native GPLs are still popular in fintech, though it requires developers to manually write complex Indian-specific logic to perform tax and audit operations. This results in inconsistent, buggy, and non-compliant code. A DSL will be able to offer high-level, terse expressions of common financial transactions, improving productivity, precision and reliability.

### *Domain-Specific Needs in Indian Finance and Commerce*

The financial system of India is strongly controlled by the RBI and GST Council, as well as, NPCI (National Payments Corporation of India). Each state and industry has compliance regulations and the environment is labyrinthine. This can be facilitated by a DSL that provides built-in functions and data types that represent Indian finance which reduces the technical obstacles to compliant software.

### *Importance of Regulatory Compliance in Financial Applications*

The first step is to comply immediately to prevent the penalties and save the reputation. Financial software has to be safe and verifiable to settle instantly, detect fraud and reconcile ledger entries. The complex regulatory checks can be substituted with a simple, high level command with a DSL such as the eTamil, accelerating the development process and limiting errors in any company of any size.

### *The Role of Language Accessibility and Localization*

Native languages like Tamil are more tolerated by many Indian finance professionals who are not in metros. They are left out in English only tools and therefore, eTamil is going to address this issue by offering bilingual keywords to allow Tamil Unicode and Latin transliterations, giving the language applications by the same developers and experts in the domain.

### *The Vision for eTamil DSL*

The suggested eTamil DSL will be an all-encompassing solution to the Indian financial ecosystem. Its features include:

- Domain-specific expressiveness
- Built-in regulatory support
- Bilingual keywords
- A safe language compiler written in Rust and LLVM.
- Hyperledger blockchain integration

The aim of this combination is to democratize the software development of finance in India.

## Overview Of Domain-Specific Languages (DSLs)

### *Definition and Characteristics of DSLs*

A general-purpose language like Python or Java cannot develop like a DSL because it is makeable in any problem domain. DSLs are very much expressive and productive in their own niche. Key traits are:

- *Conciseness* – express complex ideas in fewer lines.
- *Abstraction* - give high-level domain concepts.
- *Readability* - syntax is like natural language or familiar terminology.
- *Reduction in errors*- clustering of rules reduces errors in implementation.

Examples are SQL to database and HTML/CSS to web page.

### *Benefits of DSLs in Financial and Regulatory Domains*

DSLs have a number of benefits in the finance:

- *Intrinsic logic* - incorporate tax rules, formulas and validation.
- *Auditable code* – generate consistent, traceable, compliant output.
- *Reduced learning curve* - business experts need not have strong programming skills to write and read business rules.
- *Automation* – automate invoice generation and tax calculation with clear syntax.
- *Minimizing of errors* - Compile-time tests identify errors in the early stages.

eTamil will be able to provide features that generic languages cannot provide in relation to Indian finance.

### *Limitations of General-Purpose Languages for Domain-Specific Applications*

Finance is one area in which GPLs are difficult:

- *Boilerplate* – repetitive code for regulatory checks.
- *No graphical rules inbuilt-in* - complex logic has to be written manually by developers.
- *Skill gap* - finance professionals do not always know programming and vise-versa.
- *Fragmentation* - the teams can exercise inconsistent rule implementation.

A DSL solves these problems by directly incorporating domain knowledge, so it is consistent and accurate.

### *Existing DSLs in Global Financial Systems*

Available DSLs prove the worth of specially developed language in finance. They mainly cater to the markets of the west and do not support the Indian rules or localisation.

- *Murex and OpenGamma*: Risk and pricing in investment banking.
- *FpML*: European regulatory reporting with support of ISDA.
- *RegTech DSLs*: Automate regulatory compliance.

- *DAML*: A smart contract ledger DSL.

In addition to the above DSLs, there are other DSL implementations such as *Solidity* for blockchain contracts, *ABAP* for SAP (ERP), *OpenFisca* for tax modeling, and *Ledger CLI* for accounting demonstrate domain-driven design in financial computing. These examples highlight the global movement towards DSL-based automation, though none address India-specific financial regulations. These reasons indicate how a language such as eTamil would be viable and relevant to the Indian context, which would provide these benefits.

## **Regulatory And Functional Landscape In India**

### ***Taxation: GST, ITR, and TDS***

The taxation structure of India is complicated, and has numerous levels of compliance. GST entails multi-state registration, computation of the tax and reconciliation. ITR entails direct tax regulations such as deductions and exemptions. These may be automated in a DSL which checks formats and produces reports which are ready to comply.

### ***RBI Guidelines and Banking Compliance***

All the banking and payment systems are regulated by the Reserve Bank of India. Fintech sites are required to comply with its payment, wallet and KYC circulars. A DSL has the ability to incorporate KYC/AML controls, transaction limits, and report automation to provide end-to-end compliance and reduce audit risk.

### ***Fintech Standards: UPI, ULI, and AePS***

India is a pioneer in digital payment by the NPCI systems such as UPI, ULI and AePS. These platforms need to be complexly validated and settled and also need to detect frauds. These operations can be packaged in reusable functions in a DSL and have to be compliant with the NPCI standards.

### ***Customs, Excise, and Trade Regulations***

The acts of compliance with e-way bills rules and the Customs Act, 1962, among others are involved in international trade. The codes can be validated by a DSL and the custom declaration and calculation of duties can be generated automatically.

### ***Audit, Reporting, and Legal Documentation***

The Companies Act, 2013 and the ICAI preside over auditing. This involves keeping audit trail and creating reports. DSL may mechanize these procedures through blockchain-based registry systems,

and embedded templates, so that these platforms align with government portal systems such as MCA21 and GSTN.

### *Summary of Functional Requirements for eTamil*

eTamil must:

- Offer Accounting in Indian standards & principles as per ICAI guidelines.
- Support Indian tax laws (GST, ITR).
- Include syntax for RBI and KYC guidelines.
- Provide abstractions for UPI and other fintech platforms.
- Handle customs and trade operations.
- Offer audit-ready structures and reporting.
- Support multilingual syntax for accessibility.
- Integrate blockchain for security and traceability.

Such needs are the fundamental competencies of the eTamil language.

## **Literature On Indian Language Computing And Localization**

### *Tamil Language Computing: Unicode and Transliteration*

Tamil is a classical language with a native population of more than 80 million. It has a good digital presence because of extensive use of Unicode. Programming languages with native Tamil syntax are still mostly experimental, though typing in Tamil can be done with tools such as NHM Writer and Google Input Tools. The main opportunity and difficulty of a DSL is the development of a unified system of transliteration between Latin and Tamil scripts of the keywords.

### *Multilingual Interfaces in Software Development*

Conventional software localization concentrates on user interfaces, messages and documentation, rather than on the programming language itself. Encouragement of localization of programs has been very limited, although India has efforts such as BOSS Linux to localize operating systems, programming languages are seldom localized, which restricts the involvement of non-English speakers in core software development. The Indian programming languages like Ezhil, Niral, Bhailang are experimental and to teach programming, but not for production. Efforts in academia, including the Tamil-based educational platform Swaram, show viability but without a strong compiler and cross platform support needed to be used in the industry.

## Accessibility and Adoption of Native Language Programming

Studies have shown that native language interfaces are better in learning and usability, particularly in technical field. In rural and semi-urban India, with a larger population of professionals more at ease with Tamil, the English-based programming environments serve as a major obstacle; eTamil seeks to reduce this barrier by allowing domain experts to provide direct input in the software development process and thus increase the accuracy of business rules. This is in accordance with the National Education Policy (NEP) 2020 of India that gives a preference to regional languages in higher education.

### Existing Tools and Challenges in Localization

Although there is some progress, there are still problems. Tamil schemes of transliteration lack standardization. The problems of legacy encodings, inconsistent fonts, and lack of support of modern IDEs and debuggers are further obstacles. These issues outline the necessity of an integrated language and toolchain that is not only technically sound, but linguistically precise as well.

Standards for Tamil typing with Latin fonts					
க	ங	ச	ஞ	ட	ண
k	w	c	W	t	N
த	ந	ப	ம	ய	ர
q	n	p	m	y	r
ல	வ	ழ	ள	ற	ன
l	v	z	L	R	Z/n
அ	ஆ	இ	ஈ	உ	ஊ
a	A	i	I	u	U
எ	ஏ	ஐ	ஒ	ஓ	ஔ
e	E	Y	o	O	V
ஃ	ஹ	ஐ	சுடி	ஷ	ஸ
h	H	j	x	S	s

### Toward a Dual-Script DSL Design

eTamil suggests a dual-script system in order to make it the most accessible. Both the Tamil Unicode variant and the Latin transliteration in a new standard of matching Latin to Tamil alphabets, which enables a flexible approach to the learning process and enables cross-user interaction. It also makes it compatible with the available development tools which may not support Tamil scripts. This is a two-script model which provides a special treatment of a financial DSL.

### Comparative Features of eTamil with other DSLs

Feature / Framework	OpenFisca	ABAP	Solidity	eTamil
Domain	Tax	ERP	Blockchain	Finance, Tax, RBI, UPI
Language Localized	No	No	No	Tamil + Latin
Indian Compliance	No	Partial	No	Yes
Blockchain Support	Limited	No	Yes	Yes
Regulatory Alignment	No	Partial	No	Full

## Compiler And Language Design In DSLs

### *Why Rust for DSL Compiler Development?*

**Rust** is a high-performance language, safe and concurrent. It is best suited to construction of the eTamil compiler because it has all the features necessary to build a compiler such as a memory safety with no garbage collection, an elaborate type system to enforce domain constraints and intrinsic concurrency. The contemporary package ecosystem and easy integration with LLVM allow building of efficient and production-ready tools.

### *LLVM as the Backend Compiler Framework*

LLVM is a flexible compiler architecture, and as the back-end to eTamil, offers cross-platform portability (Linux, Windows, WebAssembly), machine-code optimization in real-time financial applications, and comprehensive tools to debug and perform static analysis. Tradeoffs Rust is expressive, combined with the powerful LLVM backend to provide a fast, reliable, and portable eTamil compiler.

### *Language Design Considerations for eTamil*

The design of eTamil aims at two objectives, and those are the domain expressiveness of financial operations and the linguistic accessibility of the Tamil language. Key design elements include:

- *Lexical organization:* A clear, bidirectional mapping between Tamil Unicode and Latin transliterations for keywords and identifiers.
- *Built-in financial functions:* Built-in income tax, GST and so on calculators like `varumAnavari()` and `jiesti()`.
- *Dual-script syntax:* The language allows Tamil scripts together with Latin. For example,

*வரவு > 500000 எனில் { ... }*

is equivalent to

*varavu > [?]500000 enil { ... }*

This two-script solution brings a new model to a financial DSL.

### *Domain-Specific Parsing and Semantic Analysis*

The eTamil compiler works by a concise pipeline:

- *Lexical Analysis:* Detects tokens in both Tamil and Latin scripts.
- *Syntax Parsing:* Construction of a grammar tree with parser generators.
- *Type Checking:* Checks operation, e.g. by not allowing combinations of illegal types of money.
- *Semantic Rules:* Imposes rules of regulation, like checking that GST rates are legal.
- *Intermediate Representation:* Compiles the program into LLVM IR to allow optimisation.

- *Code Emission*: Generates native binaries or WebAssembly. With annotated rules of compliance, the compiler is able to find mistakes during compilation.

### *IDE and Tooling Support*

eTamil will include a powerful Integrated Development Environment (IDE) to promote adoption and this will include:

- Syntax highlighting for Tamil and Latin scripts.
- IntelliSense and autocomplete which propose financial words.
- Error messages are shown in English and Tamil.
- Ready-made templates on tax and audit reports.

The IDE will be connected to external APIs- GSTN and NPCI to assist in real-life financial activities.

## **Review Of Existing Literature And Related Work**

### *DSLs in Accounting and Finance*

*QuantLib* and *Murex* are financial DSLs, which make their risk modelling easier. Nonetheless, they aim at the international markets and do not support the specific tax and compliance regulations in India - GST and ITR.

### *Regulatory Technology (RegTech) DSLs*

RegTech DSLs, such as *RegEL* and *OpenFisca* convert law into code. They are, however, specific to jurisdiction (e.g. the EU or France) and difficult to apply to the complicated Indian regulatory context.

### *Language-Driven Development in Non-English Contexts*

Efforts have been made to create software in the local languages. Indian-language computing is possible, and the *Indic Project* and learning tools like *Ezhil*, *Swaram* prove that it is possible. However, such projects typically involve education or basic utility and lack a multifaceted, compliance-oriented programming language, audit and fintech features.

### *Limitations of Current Research*

Current financial DSLs and local-language initiatives have a number of significant gaps that eTamil is aimed at addressing:

- *Indian Noncompliance*: There is no direct language to describe Indian regulations.
- *Low Level of Language Accessibility*: There are not many tools that combine Tamil Unicode and Latin transliteration, which restricts bilingual coding.

- *Lack of Integration:* The existing tools lack an integration with key Indian APIs including GSTN and NPCI.
- *None, No blockchain integration:* No DSL is connected to auditable financial workflows through blockchain systems such as Hyperledger in India.
- *Partial Solutions:* In use are libraries or micro-languages, which do not provide a complete back-end server or database.

## **Research Gap And Justification For eTamil DSL**

### ***Lack of Indian Regulatory-Integrated DSLs***

Spreadsheets and general purpose languages are not designed to support the Indian financial and regulatory systems. They compel users to enter large amounts of manual code to perform basic operations, GST, ITR forms, RBI guidelines - which slows down the automation process, reduces program efficiency, and increases the possibility of error.

### ***Inaccessibility for Non-Programmers and Regional Professionals***

Indian finance experts are many but not all of them have the programming skills. Broad-scope languages may be daunting, and are linguistically foreign, which could be a significant barrier, eTamil bridges this barrier by providing experts with a way to express financial logic in Tamil keywords and a dual-script system that works with Unicode and Latin transliteration.

### ***Inadequate Linguistic and Compliance Localization in Existing Tools***

Past India-based localization had usually focused on user interfaces, ignoring syntax, compiler structure. Accordingly, fundamental tax and banking principles are not coded as language primitives. In addition, the non-existence of a bilingual coding standard is an obstacle to mass adoption.

### ***No Full-Stack DSL for Finance-Compliant Backend Development***

An Indian financial system must not have just a rule representation, but database connectivity, integration with government APIs (ex: GSTN), and secure processing of transactions, preferably through blockchain. None of the existing DSLs provide such a complete, full-stack solution targeted at the Indian regulations, eTamil fulfils that gap as it provides a full back-end development toolkit.

### ***Justification for eTamil DSL***

*eTamil* is justified with its special contributions:

- *Domain Specificity*: specific to the trends of Indian finance, accounting, and fintech.
- *Regulatory Awareness*: Adds GST, ITR, RBI, UPI and customs abstractions.
- *Linguistic Inclusivity*: Provides Tamil-language keywords in both Unicode and Latin scripts.
- *Technological Modernity*: Safely and high-performance built using Rust and LLVM.
- *Accessibility*: Facilitates co-operation between programmers and domain experts.
- Also *Digital India Alignment*: Facilitates the national push towards digital governance and strengthens regional languages.

eTamil is not just a scholarly work of language and code but also a political instrument that will democratize the process of fintech development and regulatory automation throughout India.

## Proposed DSL Framework — eTamil

### Design Philosophy

eTamil is designed on five principles:

- *Domain-Centric Abstractions*: Focuses on Indian finance and compliance.
- *Linguistic Accessibility*: Provides Tamil keywords in Unicode and Latin.
- *Regulatory Awareness*: Embeds logic for GST, ITR, RBI, UPI, and customs.
- *Full-Stack Capabilities*: Provides backend services, databases, APIs and blockchain tools.
- *Modern Infrastructure*: Uses Rust and LLVM for performance and extensibility.

### Language Syntax and Keywords

eTamil accepts two keying forms, Tamil Unicode (e.g., வருமானவரி), and Latin transliteration (e.g., varumAnavari). This two-script format contains keywords that are explicit and precise and case-sensitive. As an example, one of the simplest tax calculations would be as follows in eTamil:

```
வருமானம் = 800000
வரி = வருமானம் * 0.1
அச்சு(வரி)
```

Or, in Latin script:

```
varumAnam = 800000
vari = varumAnam * 0.1
accu(vari)
```

### Compiler and Toolchain Architecture

The eTamil compiler works via a pipeline starting with a lexing phase which can process dual-script keywords, then a syntax parser written in Rust libraries and a semantic analysis phase which implements domain constraints like GST slabs. Then the compiler generates LLVM IR to optimize

and generate code. The full toolchain includes a linter, an interactive experimentation REPL, and a plug-in architecture.

### *Framework Modules*

eTamil ecosystem will offer modular structures of major domains:

- *Accounting* (ledgers, transactions),
- *Taxation* (GST/ITR/TDS templates),
- *Banking & RBI Compliance* (NEFT/UPI), and
- *Customs & Excise*.

Also, it will be having a Blockchain Integration module to provide audit trails and Database Support module with an ORM-like feature to provide ledger and compliance logs.

### *Example Use Cases*

eTamil simplifies complex tasks. To demonstrate, it can produce GST Returns with one call to a function, automate TDS Deduction, reconcile UPI Payments and produce a tamper-proof Blockchain Audit Trail using built-in templates and logic.

### *Standards and Interoperability*

To ensure that the adoption and retention is non-disruptive:

- *Rules of Latin Transliteration*: We are going to develop a standardisation of Tamil script to Latin characters, which is different from ISO 15919.
- *Unicode Compliance*: The Tamil script tokens will be in full compliance to Unicode norms.
- *REST API Bindings*: eTamil programs can be interrelated with GSTN, NPCI and government portals.
- *Cross-Language Interop*: LLVM bytecodes generated to interoperate with Rust, Python or R-based systems.

### *Security, Privacy, and Legal Compliance*

eTamil shall abide by:

- *Data Protection Rules*: Fields with encrypted fields and audit logs.
- *Digital Signature Integration*: For invoice verification and legal submissions
- *Compliance Hooks*: Validation against MCA, RBI, SEBI, and GST norms
- *Blockchain Ledger Immutability*: Leveraging Hyperledger for audit-proof records

## Conclusion And Future Directions

### *Summary of Findings*

This review has looked at why a DSL is required to serve the financial and fintech sector in India. The analysis shows that the current general-purpose languages do not have native support of India-specific regulations such as GST, ITR and RBI compliance. The proposed eTamil DSL would fill this gap with the inherent abstractions and the dual-script language approach in Tamil and Latin transliterations, which makes it more accessible and corresponds to the Digital India mission. Both technically the project is viable with the use of Rust and LLVM to develop a compiler that will be scaled and modules will be included to assist in taxation, banking and blockchain.

### *Potential for Policy Adoption and Digital Governance*

eTamil has important implications to public policy. It helps in the Digital India project by fostering the development of technology in the Indian languages and automation of regulation. It has the potential to boost financial transparency and as a regulatory sandbox of fintech. Government departments and educational institutions might also adopt it to help close the technology and finance disjuncture in skills.

### *Roadmap for Full Implementation and Community Adoption*

eTamil will be a multi-phase implementation.

- *Phase 1* is concerned with the compiler and core language development.
- *Phase 2* will entail the development of domain specific modules in accounting, taxation and banking.
- *Phase 3* will develop tooling, a REPL shell and database integration.
- *Phase 4* will be pilot projects and open-source release.
- Lastly, *Phase 5* will be dedicated to policy involvement with major government entities such as the MCA, RBI and GSTN.

## Final Note

Through integrating the language of law, finance and code, eTamil can transform the way financial software is created in India by offering an all-encompassing and future-proven approach to the digital economy.

## References

Meacham, Sofia. "Evaluation of Classification Algorithms Framework Domain-Specific Language: the case of finance-accounting domain." 2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC). IEEE, 2023.

Voelter, Markus, et al. "A Domain-Specific Language for Payroll Calculations: A Case Study at DATEV.", voelter.de, 2020.

Doshi, Janhavi. "A Domain-Specific Language for Accounting", Rochester Institute of Technology, 2024.

Ramautar, Vijanti, and Sergio España. "The OpenESEA modeling language and tool for ethical, social, and environmental accounting." *Complex Systems Informatics and Modeling Quarterly* 2023.34: 1-29, 2023.

Ungureanu, Vlad, et al. "A dsl solution for Moldova's tax system." *Conferința tehnico-științifică a studenților, masteranzilor și doctoranzilor. Vol. 2.* 2024.

Merigoux, Denis, Raphaël Monat, and Jonathan Protzenko. "A modern compiler for the french tax code." *Proceedings of the 30th ACM SIGPLAN International Conference on Compiler Construction.* 2021.

Mirković, Aleksa, et al. "A model-driven approach to establishment of private blockchain business networks." *Proceedings of the 9th International Conference on Information Society and Technology. ISOS Conference Proceedings Series.* 2019.

Olivieri, Luca, et al. "General-purpose Languages for Blockchain Smart Contracts Development: A Comprehensive Study." *IEEE Access*, 2024.

Dhanya, V. R., Rivika Richard D'silva, and David Joseph. "Regulatory Challenges and Compliance in Decentralized Finance (DeFi): Comparative Study Between India and USA." *Machine Learning and Modeling Techniques in Financial Data Science*. IGI Global Scientific Publishing: 71-100, 2025.

Ganesh, S. G., G. R. Prakash, and KK Ravi Kumar. "An Overview of 'Swaram': A Language for Programming in Tamil." *Infitt. Net*: 96-103, 2003.

Raj, Adalbert Gerald Soosai, et al. "What Do Students Feel about Learning Programming Using Both English and Their Native Language?." *2017 International Conference on Learning and Teaching in Computing and Engineering (LaTICE)*. IEEE, 2017.

Parambil, Adithya Jayakumar, et al. "Saarthi: A programming language designed to introduce coding to high schoolers." *2024 11th International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2024.

Annamalai, Muthiah. "Ezhil: A Tamil Programming Language." *arXiv preprint arXiv:0907.4960*, 2009.

Sunayana, Bhumireddy, et al. "Survey of Non-English Language Compilers: (Exploring the Diversity of Programming Languages)." *2023 9th International Conference on Advanced Computing and Communication Systems (ICACCS)*. Vol. 1. IEEE, 2023.

Kulkarni, Vinay, et al. "Toward automated regulatory compliance." *CSI Transactions on ICT* 9.2, 2021.

Suresh, R. "The future of risk management in Indian Banking: Aligning with fintech and regulatory innovation." *International Journal of Literacy and Education*, 2024

Routray, Susmi, and Reema Khurana. "A Study of Information Communication Technology and e-Governance Initiatives in India." *International Journal of Arts and Sciences* 3.12: 152-164, 2010.

*Miller, Eluide, et al. "Success of a Customs and Excise Department Information System: A Developing Nation Perspective." First Annual Research for National Development Conference, University of Belize, 2017*

*Al Qudah, Anas, Lara ALhaddad, and D.U.N.Y.A. Khaled Elwaked. "Financial technology as an anti-corruption tool: a review of blockchain, AI, And RegTech Applications." Journal of Tianjin University Science and Technology 58.06, 2025.*