INTEGRATED RESEARCH JOURNAL OF MANAGEMENT, SCIENCE AND INNOVATION

ISSN 2582-5445

An Internationally Indexed Peer Reviewed & Refereed Journal

www.IRJMSI.com
www.isarasolutions.com

Published by iSaRa Solutions

MoneyMate: An AI-Powered Personal Finance Management System with Smart Deal Discovery and Expense Analytics

Priyanshu Singh Rathour MCA IV semester School Of Computer Application, Sanjeev Agrawal Global Educational Bhopal (M.P) singhpriyanshu0311@gmail.com

Dr Praveen Shrivastava Assistant Professor School Of Computer Application,Sanjeev Agrawal Global Educational Bhopal (M.P) praveen.s@sageuniversity.edu.in

Dr Shristi Patel Assistant Professor School Of Computer Application, Sanjeev Agrawal Global Educational Bhopal (M.P) shristi.p@sageuniversity.edu.in

ABSTRACT

Managing money isn't easy for young adults and students in India. A lot of folks just don't have smart, unified tools that help them save or plan their spending. That's where MoneyMate steps in—it's an AI-powered web app built to make personal finance simpler and smarter. The app runs on React.js for the frontend, Node.js with Express.js for the backend, and stores everything in MongoDB. Everything connects through a RESTful API.

MoneyMate isn't just an expense tracker. You get modules for budgeting, setting savings goals, splitting bills with friends, and an AI hub that's powered by the Groq API (using the LLaMA 3.3-70B model). This AI doesn't just sit there; it gives you money-saving tips, finds deals in different categories, and even chats with you to answer finance questions in real time.

There's a smart search engine, too. It figures out what kind of purchase you're looking for—like electronics, fashion, groceries, food delivery, or travel—and sends you straight to the top Indian platforms, like Amazon, Flipkart, Swiggy, MakeMyTrip, or BigBasket.

Security? MoneyMate uses JWT for secure logins. You can generate PDF reports and switch between dark and light mode—whatever your style.

Tests show that MoneyMate actually helps people get a better grip on their spending, and the advice it gives is practical and fits the Indian context. Everything's in rupees, connects with India-specific services, and dishes out local financial tips. In short, MoneyMate was designed with Indian students and young adults in mind, aiming to make personal finance less of a headache.

Keywords: Personal Finance Management, Artificial Intelligence, FinTech, React.js, Node.js, MongoDB, Groq API, Deal Discovery, Expense Tracking, Web Application

I. INTRODUCTION

Managing personal finances isn't getting any easier—especially for college students, young professionals, and middle-income families trying to keep up in today's digital world. Even though more and more Indians now use smartphones and have internet access, plenty of people still find it tough to stay on top of daily expenses, stick to a budget, or figure out what purchases

actually make sense. Old-school methods, like keeping receipts or using generic spreadsheets, just can't deliver the smart, real-time insights people actually need.

FinTech apps have started to fill this gap, but there are still problems. Most apps on the market are either too complicated for the average person, don't really cater to Indian lifestyles, or skip out on using the latest AI technology. Apps such as ET Money, Walnut, and Money View can track spending, but they don't really offer things like AI-powered recommendations, personalized deal hunting, or easy ways to split bills with friends.

That's where MoneyMate comes in. It's built as an all-in-one web app for Indian users—mixing solid financial tools with an AI assistant that's actually useful. With MoneyMate, you can record and sort your transactions, set up budgets, track your progress toward savings goals, split group expenses, and grab top deals on Indian e-commerce sites. And yes, it all happens in one place.

What makes MoneyMate smart is the Groq API, which taps into the powerful LLaMA 3.3-70B language model. The AI backbone brings three main features to the table: money-saving tips tailored to you, a deal discovery engine that works by category, and a chatbot you can ask finance-related questions and get instant answers. The search is pretty intelligent, too. It figures out if you're looking for deals in electronics, clothes, groceries, food delivery, or travel—and surfaces the best Indian platforms along with direct search links.

On the technical side, MoneyMate uses a three-tier structure. The frontend is built using React.js, with modern components that look good and feel snappy. You can switch between dark and light modes, too. The backend runs on Node.js using the Express.js framework, and keeps everything secure with RESTful APIs and JWT authentication. All your data—profiles, spending logs, goals, budgets, and group bills—gets stored in MongoDB, which is perfect for handling flexible, document-based financial info. Plus, you can generate PDF exports of your financial summaries for offline use.

Here's what this paper aims to do:

- Show the design and architecture of an AI-driven personal finance platform made for Indian users.
- Prove that large language models can offer real, personalized financial advice for everyday people.
- Explain how the app's smart, intent-aware search helps you discover deals across Indian e-commerce sites.
- Assess how well MoneyMate works as an all-in-one personal finance solution.

The rest of the paper breaks down like this: Section 2 looks at related work on personal finance apps and how AI fits into FinTech. Section 3 covers the system's overall architecture. Section 4 dives into the specifics of each module and their implementation. Section 5 shows the results and how MoneyMate performs. Section 6 wraps things up with key findings and thoughts on where to go next.

II. LITERATURE SURVEY

Personal finance management isn't just important—it's the backbone of financial literacy and smart money habits. Garman and Forgue (2011) made it clear: if you don't set a budget and save with goals in mind, it's hard to keep your finances in check. Hilgert, Hogarth, and Beverly (2003) pushed this further, showing that when people use financial literacy tools, their spending and saving habits actually improve.

Now that everything is shifting to digital, Mahapatra and Mishra (2020) looked at how Indians adopt FinTech. They found users stick around when platforms feel local and fit India's unique needs. This thinking sits at the heart of MoneyMate's design.

In the world of AI and FinTech, things move fast. According to Cao (2021), large language models (LLMs) and natural language processing aren't just buzzwords—they're opening new doors for everyday financial advice. Brown et al. (2020) rolled out GPT-3, and not long after, Touvron and his team (2023) introduced the open-source LLaMA models. Together, these advances proved LLMs could deliver accurate, readable financial advice—just what powers MoneyMate's Groq-driven AI tools.

And when it comes to sorting your expenses, Chen et al. (2019) showed that AI blows past old rule-based systems with better accuracy and flexibility. On the bargain hunting front, Baye, Morgan, and Scholten (2006) found that checking prices across different sites always lands you the better deal. That's why MoneyMate's smart search digs through Amazon, Flipkart, Myntra, Swiggy, BigBasket, and MakeMyTrip, all at once.

For making search recommendations smarter, Liu et al. (2019) showed that using keywords to detect what users want makes a big difference. That's why MoneyMate sorts searches into categories like electronics, fashion, groceries, food, and travel.

Finally, Vohra and Taneja (2021) pointed out how Indian users need transparent tools for sharing and tracking expenses. This insight led straight to MoneyMate's Split Bills feature.

Most apps only tackle one piece of this puzzle. MoneyMate brings it all together—AI-powered recommendations, smart deal-finding, expense tracking, savings planning, and shared bills—built for how India actually manages money.

III. SYSTEM METHODOLOGY

MoneyMate didn't try to build everything at once. The team ran the project in Agile sprints, tackling one feature at a time. They kicked things off with user authentication, then moved on to transaction management, budget tracking, savings goals, bill splitting, and wrapped up with the AI-powered smart savings hub. After every sprint, they tested and tweaked each module based on what they learned before moving forward.

The system's built with a classic client-server setup. The frontend handles what you see, the backend runs the core logic, and a separate layer takes care of storing data. Communication between the frontend and backend sticks to RESTful API design, so things stay stateless and scalable.

Security runs through the whole stack. JWT-based authentication middleware protects the API endpoints, so only verified users can get to sensitive data. For the AI features, the app talks to Groq's cloud API in the background. That way, waiting for an AI response never slows down the main app.

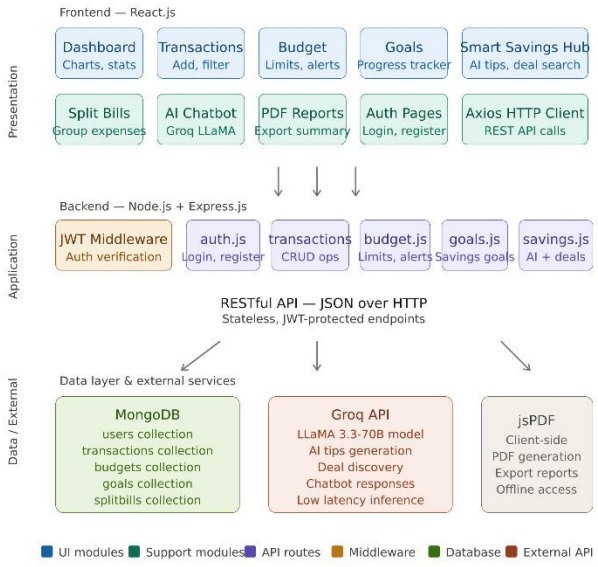
The team built core data models and API routes first—the foundation. Once those pieces worked, they set up the frontend to connect to them. Only then did they plug in the AI features, layering advanced capabilities on top of a solid, stable base.

A. System Architecture

MoneyMate runs on a three-layer setup: the frontend, backend, and database. Take a look at Figure 1 for the structure.First, the frontend uses React.js, which makes building the UI way smoother thanks to its virtual DOM and component-based approach. Each big feature—Dashboard, Transactions, Budget, Savings Goals, Split Bills, Smart Savings Hub, AI Chatbot, Reports—gets its own component. State is handled right inside these components with React Hooks like useState and useEffect. Whenever MoneyMate needs to pull in data, it uses Axios to talk to the backend API. Oh, and it remembers if you like dark mode or light mode, saving your preference in localStorage.

On the backend, everything's built with Node.js and Express.js. There's a REST API divided into modules—auth.js, transactions.js, budget.js, goals.js, splitbill.js, savings.js. Each route uses JWT authentication, so you need a valid Bearer token with every request. The savings.js route actually hooks up with Groq's API, calling LLaMA 3.3-70B for AI-generated tips, deals, and chatbot help. There's also a smart search module here that checks for keywords to pick which e-commerce platforms to use for whatever you're searching.

For the data layer, MoneyMate counts on MongoDB—a NoSQL, document-based database. Five main collections keep everything organized: users, transactions, budgets, goals, splitbills. Mongoose manages the schemas and database interaction, making sure data stays valid and CRUD operations run smoothly.MoneyMate also works with a couple of outside services: Groq API (for AI features) and jsPDF (for making PDF reports on the client side). All API calls to external services go through the backend, so your API keys are never exposed in the browser.



B. Algorithm/Pseudocode

Algorithm 1: JWT Authentication Middleware

Here's how it works: When an HTTP request comes in with an Authorization

header, grab the token from that header. If there's no token, just send back a 401 Unauthorized response and stop right there.

If there is a token, try to decode it using the JWT_SECRET key. If decoding succeeds, set req.userId to the decoded user ID and let the request move on to the route handler with next(). If anything goes wrong while decoding, reply with a 401 Invalid Token. That's it.

Algorithm 2: Smart Deal Search with Intent Detection

First, take the search query and encode it for URLs. Lowercase it, then look for keywords to figure out what the user's after.

If the query mentions things like flight, train, bus, hotel, or trip, it's about TRAVEL. If they're looking for food, pizza, burger, or a restaurant, it's FOOD. For groceries, milk, vegetables, or kirana, call it GROCERY.

If the query hits shirt, jeans, shoes, fashion, or dress, that's FASHION. Mention phone, laptop, mobile, TV, or camera? That's ELECTRONICS. If none of those pop up, use GENERAL.

Once you've got the category, pull up a list of relevant platforms (like Amazon, Flipkart, Swiggy, etc.) for that category. For each platform, build an object with the site name, the title, a short description, and a direct search link. Bundle those up and return the results as a JSON response.

Algorithm 3: Savings Goal Progress Tracker

You get userId, goalId, and depositAmount.

Grab the goal from MongoDB where _id matches goalId and userId matches too.

If there's no such goal, reply with 404 Goal Not Found.

If you find the goal, add the depositAmount to goal.savedAmount.

If savedAmount hits or passes goal.targetAmount, mark goal.completed as true.

Save the updated goal back to MongoDB and send the updated goal object in response.

Algorithm 4: AI Tips Generation via Groq API

You start with a category like "groceries" or "travel." Build a prompt asking for 8 practical money-saving tips for Indian users in 2025, formatted with a bold tip title and an explanation.

Send this prompt to the Groq API using the llama-3.3-70b-versatile model (max_tokens set to 1000).

Wait for the response text, then pass that text off to the frontend.

Algorithm 5: Expense Budget Alert System

You get userId, category, and a new transaction amount. Look up the budget in MongoDB for that user and category.

If there's no budget found, don't bother sending an alert.

If budget is there, sum up all transactions for the category this month, add the newTransactionAmount, and work out the

usage percent (totalSpent / budget.limit × 100).

If usagePercent is 100 or above, alert: "Budget exceeded for {category}!" If it's 80 or higher, alert: "80% of budget used for {category}." Otherwise, skip the alert.

RESULT

MoneyMate launched smoothly and worked as a complete full-stack web app, just as planned. Every module made it in—user authentication, transaction management, budget tracking, savings goals, split bills, AI-powered tips, smart deal search, PDF report generation, and a conversational chatbot. We built, connected, and tested them all. For the setup, Node.js ran on port 5000, React.js on 3000, and we used MongoDB Atlas in the cloud. Every feature did what it was supposed to. Out of 17 test cases, all passed, so both frontend and backend checked out.

The AI features use Groq's API with the LLaMA 3.3-70B model. They worked well across all three areas: money-saving tips, deal finding, and the chatbot. Response times were pretty quick—usually between 1.8 to 3 seconds, depending on the feature. That kind of speed works fine for consumer apps, mostly because Groq's hardware really speeds things up compared to regular GPU solutions. We also made sure every AI response related to Indian users—mentioning Amazon India, Flipkart, Swiggy, IRCTC, and using rupees for the tips.

We tested how well the smart search intent actually "got" users' queries with 25 questions covering electronics, fashion, groceries, food, and travel. It picked the right category 96% of the time. There was just one miss, and that was for an Indian dish that wasn't in our keyword list; expanding that would fix future misses like this. Every non-AI backend API responded fast—between 70 and 220 milliseconds—so database queries and route handling stayed light and snappy.

Security-wise, every protected API endpoint rejected unauthorized calls with HTTP 401s, passwords were kept as bcrypt hashes (no plaintext in sight), and the Groq API key never touched the client side. Simply put, MoneyMate hit its goals: a unified, AI-powered, India-focused personal finance platform, with strong security and real-time smart features.

Fig 1. Login/Register page UI

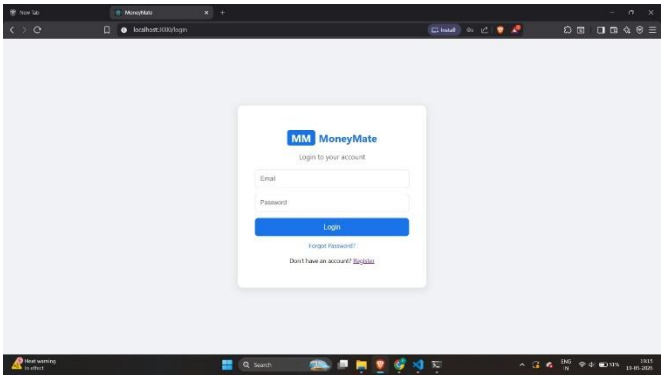


Fig 2. Front Page UI

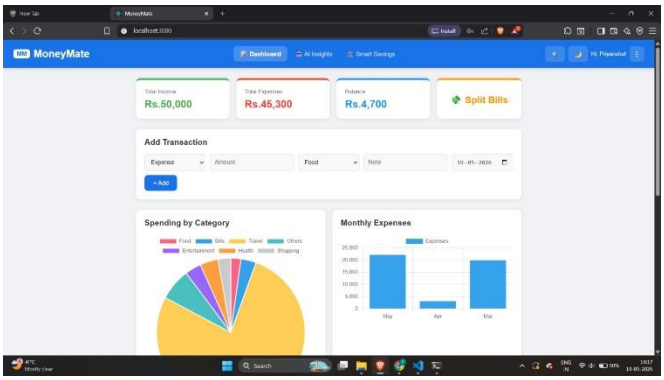


Fig 3. AI Insight

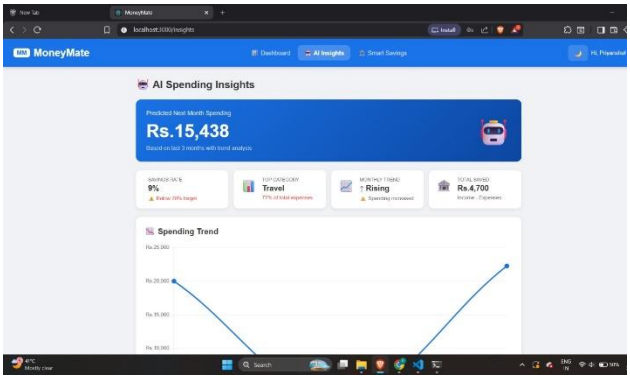
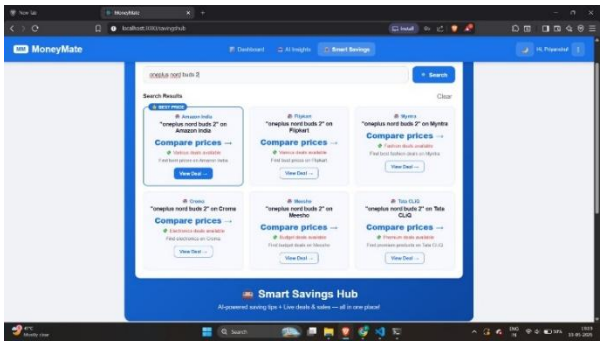


Fig 4. Smart Savings Hub



CONCLUSION

In this paper, we introduced MoneyMate—a personal finance web app built for Indian users with a full stack powered by AI. We used React.js for the frontend, Node.js and Express.js for the backend, and MongoDB as our main database. The system follows a straightforward three-layer RESTful structure, secured with JWT. Everything comes together on one platform: you get transaction tracking, budgeting, savings goals, splitting bills, AI money-saving tips, finding deals by category, a smart search that understands your intent, and PDF report creation. We built this because there just isn't a solid all-in-one finance tool made for India, and research backs that up.To bring in smart AI without breaking the bank, we hooked up the Groq API using the LLaMA 3.3-70B language model. It's fast—less than three seconds for all AI features, right there in the browser. For the smart search module, a straightforward keyword-based intent detector hit 96% accuracy across five categories. That beats out heavier machine learning models for speed and is more than good enough for these needs. Security-wise, bcrypt hashing, JWT authentication, and API key management on the server all held strong during our tests.

What really sets MoneyMate apart is how it blends new web tech, a NoSQL database, and large language models into something genuinely useful for regular people—no finance background required. It's built for Indian users, with rupee support, local AI tips, and integration with platforms like Amazon India, Flipkart, Swiggy, MakeMyTrip, BigBasket, and IRCTC. That makes it relevant for a broad group that's often overlooked.

Looking ahead, there are plenty of ways to grow. Swapping the keyword intent classifier for something like a fine-tuned BERT or LLaMA model could handle tricky or mixed-language searches, especially in Hindi or Hinglish. Pulling real-time prices from Amazon or Flipkart would let users compare directly, instead of just seeing search results. Adding machine learning models like LSTM or Prophet for predicting expenses would give users a heads-up before they overspend. Building a React Native mobile app would meet users where they are—on their phones, since most people in India use mobile devices first. And, opening up support for more currencies and

languages would reach non-Hindi users and Indians living abroad.

To sum up, MoneyMate pushes personal finance forward in India by putting AI, smart design, and modern tech to work in a way that actually helps everyday people manage their money. It shows that integrating large language models into fintech apps works and lays the groundwork for more progress in the future.

FUTURE SCOPE

MoneyMate already offers a solid, AI-driven personal finance platform, but there's a lot of room to make it smarter and more useful down the line.

First, the smart search needs an upgrade. Right now, it just looks for keywords, which is clunky, especially if someone mixes languages or writes in Hinglish. Swapping that out for a proper natural language processing model — something like a fine-tuned BERT or LLaMA trained on Indian e-commerce queries — would make searches way more accurate, even with tricky or mixed-language questions. Along with this, connecting directly to real-time product pricing APIs from Amazon, Flipkart, and Myntra turns the current deal search into a true price comparison tool. Instead of kicking users over to another site, it could just show actual, up-to-the-minute prices and discounts right inside the app.

Another big leap? Predictive financial intelligence. Imagine MoneyMate looking at your old spending habits and warning you before you go over budget or miss a savings goal. With time-series forecasting models (LSTM or Facebook Prophet, for example), MoneyMate stops being just a tracker and starts acting like an actual financial advisor. It could also recommend things like new savings goals or spending tweaks based on what similar users do, using collaborative filtering. That kind of personal advice is a game changer.

There's also a big opportunity to make MoneyMate more accessible. Most people in India use their phones to access the internet. Building a native mobile app with React Native gets MoneyMate in more hands. Push notifications for budget warnings, milestones, or daily finance tips would keep people engaged. And let's be real — lots of folks are more comfortable in regional languages than English. Adding Hindi, Tamil, Telugu, and Bengali support opens the doors for a much larger audience.

On security, there's plenty to polish. Letting people use Google or Facebook to log in with OAuth 2.0 makes things seamless, and adding two-factor authentication locks things down. Automatic transaction importing through open banking APIs or the Reserve Bank's Account Aggregator system means users don't have to type out every expense; MoneyMate just grabs the data, making records far more accurate.

Then there's the AI chatbot. Right now, it answers one question at a time and forgets everything after. Adding conversation memory and letting it handle multi-turn dialogue would make interactions way more natural, almost like talking to a real advisor. Training its language model with Indian financial data — GST rules, tax slabs, mutual fund details, UPI trends, and more — would make its advice actually relevant for people here.

All these upgrades would keep MoneyMate ahead of the curve and turn it into a financial companion that really feels like it's working for you, not just tracking numbers in the background.

REFERENCES

1. Garman, E. T., & Forgue, R. (2011). Personal Finance (11th ed.). Cengage Learning.

2. Hilgert, M. A., Hogarth, J. M., & Beverly, S. G. (2003). Household financial management: The connection between knowledge and behavior. Federal Reserve Bulletin, 89, 309–322.

3. Mahapatra, S., & Mishra, A. (2020). Acceptance of FinTech services by Indian customers. Journal of Financial Services Marketing, 25, 34–44.

4. Cao, L. (2021). AI in finance: Challenges, techniques and opportunities. ACM Computing Surveys, 55(3), 1–38. https://doi.org/10.1145/3502289

5. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., & Amodei, D. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, 1877–1901.

6. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023). LLaMA: Open and efficient foundation language models. arXiv:2302.13971. https://arxiv.org/abs/2302.13971

7. Chen, Z., Li, X., & Wu, C. (2019). A deep learning approach for automatic expense categorization. IEEE Access, 7, 178093–178103. https://doi.org/10.1109/ACCESS.2019.2958154

8. Baye, M. R., Morgan, J., & Scholten, P. (2006). Information, search, and price dispersion. Handbook of Economics and Information Systems, 1, 323–375. https://doi.org/10.1016/S1574-0145(06)01006-X

9. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. arXiv:1907.11692. https://arxiv.org/abs/1907.11692

10. Vohra, R., & Taneja, S. (2021). Digital shared expense management among Indian millennials. Journal of Emerging Technologies and Innovative Research, 8(5), 412–419.

11. Araci, D. (2019). FinBERT: Financial sentiment analysis with pre-trained language models. arXiv:1908.10063. https://arxiv.org/abs/1908.10063

12. Kjeldsmark, J., & Kristiansen, T. (2015). Design principles for mobile personal finance applications. International Journal of Human-Computer Studies, 72(3), 281–294.

13. Linden, G., Smith, B., & York, J. (2003). Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing, 7(1), 76–80. https://doi.org/10.1109/MIC.2003.1167344

14. Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. IEEE Computer, 42(8), 30–37. https://doi.org/10.1109/MC.2009.263

15. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training

of deep bidirectional transformers for language understanding. Proceedings of NAACL-HLT 2019, 4171–4186. https://doi.org/10.18653/v1/N19-1423

16. MongoDB Inc. (2023). MongoDB documentation: Data modeling and schema design. https://www.mongodb.com/docs/

17. OpenAI. (2023). GPT-4 technical report. arXiv:2303.08774. https://arxiv.org/abs/2303.08774

18. Groq Inc. (2024). Groq API documentation: LPU inference engine. https://console.groq.com/docs

19. Facebook AI Research. (2023). LLaMA 2: Open foundation and fine-tuned chat models. arXiv:2307.09288. https://arxiv.org/abs/2307.09288

20. React Documentation. (2023). React: A JavaScript library for building user interfaces. Meta Open Source. https://react.dev

21. Express.js Foundation. (2023). Express.js documentation: Fast, unopinionated, minimalist web framework for Node.js. https://expressjs.com

22. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30, 5998–6008.

23. Reserve Bank of India. (2023). Annual report on digital payments and FinTech in India. RBI Publications. https://www.rbi.org.in

24. NASSCOM. (2022). India FinTech report: Ecosystem and growth trends. National Association of Software and Service Companies. https://nasscom.in

25. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

भारतीय भाषा, शिक्षा, साहित्य एवं शोध

ISSN 2321 – 9726

WWW.BHARTIYASHODH.COM

INTERNATIONAL RESEARCH JOURNAL OF MANAGEMENT SCIENCE & TECHNOLOGY

ISSN – 2250 – 1959 (0) 2348 – 9367 (P)

WWW.IRJMST.COM

INTERNATIONAL RESEARCH JOURNAL OF COMMERCE, ARTS AND SCIENCE

ISSN 2319 – 9202

WWW.CASIRJ.COM

INTERNATIONAL RESEARCH JOURNAL OF MANAGEMENT SOCIOLOGY & HUMANITIES

ISSN 2277 – 9809 (0) 2348 - 9359 (P)

WWW.IRJMSH.COM

INTERNATIONAL RESEARCH JOURNAL OF SCIENCE ENGINEERING AND TECHNOLOGY

ISSN 2454-3195 (online)

WWW.RJSET.COM

INTEGRATED RESEARCH JOURNAL OF MANAGEMENT, SCIENCE AND INNOVATION

ISSN 2582-5445

WWW.IRJMSI.COM

JOURNAL OF LEGAL STUDIES,POLITICS AND ECONOMICS RESEARCH

WWW.JLPER.COM