Venkata Sai Tharun Pudi

☎+1 (989)330-1431 | ✉ tharunpudi4@gmail.com | LinkedIn | GitHub

PROFESSIONAL SUMMARY

Software Engineer with 8+ years of experience building scalable backend services and interactive front-end applications using **Python, Django, Flask, React.js, JavaScript, SQL, and modern microservices architecture**. Skilled in designing **REST APIs**, developing reusable UI components, managing relational and **NoSQL** databases, and implementing **CI/CD** pipelines for automated builds and deployments. Experienced in containerizing applications with **Docker**, orchestrating services with **Kubernetes**, and integrating cloud-native components for secure and maintainable enterprise solutions. Adept at collaborating with cross-functional teams, analyzing technical requirements, optimizing performance, and delivering high-quality software through Agile practices.

TECHNICAL SKILLS

Category	**Technologies / Tools**
Programming Languages	Python, JavaScript (ES6+), TypeScript, SQL, Shell Scripting
Backend Development	Django, Django REST Framework, Flask, FastAPI, Node.js, Express.js, REST APIs, Microservices, JWT, OAuth2, Middleware, Async Programming
Frontend Development	React.js, Redux, React Hooks, React Router, HTML5, CSS3, Bootstrap, Material UI, Tailwind CSS, SPA Development, Responsive UI
Databases	PostgreSQL, MySQL, SQLite, MongoDB, DynamoDB, Database Design, Query Optimization, Migrations, Django ORM, SQLAlchemy
Cloud & Serverless	AWS EC2, Lambda, S3, RDS, EKS, CloudWatch, CloudFormation, SNS, SQS, Elastic Beanstalk, IAM, KMS, Secrets Manager
DevOps & CI/CD	Docker, Kubernetes, Jenkins, GitHub Actions, GitLab CI, CircleCI, Terraform, Ansible, Version Control (Git, GitHub, GitLab, Bitbucket)
Messaging & Async Processing	Kafka, RabbitMQ, Celery, Event-Driven Architectures, Background Tasks
Testing & Debugging	PyTest, Unittest, Selenium, Postman, Swagger/OpenAPI, Jest, React Testing Library, API Testing, Integration Testing, Mocking
Utilities & Data Handling	Pandas, NumPy, JSON, YAML, CSV, BeautifulSoup, Logging Frameworks
Tools & Methodologies	JIRA, Confluence, Agile/Scrum, VS Code, PyCharm, SDLC, TDD, Code Reviews, Documentation Practices

PROFESSIONAL EXPERIENCE

Senior Software Engineer **May 2024 - Present**

BNSF Railway, USA

Project Description: The Claims Processing Management System developed for BNSF Railway streamlines the handling of work assignment disputes raised by unionized field employees. When an employee believes that a task, which should have been assigned to them, was erroneously given to a third-party contractor, they can initiate a claim through the system. The process begins with the submission of the claim, followed by an initial review where BNSF representatives assess its validity. If the claim is deemed invalid but the union disagrees, the system facilitates a formal discussion phase between the union and BNSF to seek resolution. Should disagreements persist, the claim can escalate to an arbitration stage. Throughout each phase, the system ensures transparency by maintaining detailed records of all actions and decisions, thereby promoting accountability and efficient resolution of disputes.

Backend Responsibilities

- Designed and implemented backend services using **Python, Django,** and **FastAPI** to support multi-stage claim workflows including submission, review, union response, discussion, and arbitration, ensuring consistent request validation, workflow routing, and audit-driven traceability across all phases.
- Built and maintained **RESTful APIs** for managing claims, user roles, workflow transitions, document attachments, and discussion threads while ensuring secure input handling, error management, and structured response schemas aligned with enterprise integration standards.
- Engineered robust **PostgreSQL** data models to store claims, workflow states, arbitration notes, communication logs, and historical actions using optimized **ORM** queries, indexing patterns, and data-access strategies that supported reliable backend performance.
- Integrated role-based access control and authentication mechanisms within backend modules to ensure secure handling of sensitive claim information while maintaining clean separation of service boundaries across internal and external user groups.
- Refactored backend utilities, service classes, and serialization logic to improve maintainability, reduce duplication, optimize response payloads, and support consistent performance across high-traffic operational workflows within the claim’s platform.

Frontend Responsibilities

- Developed React.js front-end modules using **Hooks, Redux**, and reusable UI components to support claim creation, document upload, review dashboards, discussion panels, and arbitration views, ensuring a seamless and intuitive experience for both internal teams and union members.
- Built interactive dashboard components that provide real-time visibility into claim statuses, workflow transitions, pending approvals, and historical decisions, improving user navigation and operational clarity across multiple claim processing stages.
- Implemented structured form components, validation logic, and controlled UI states enabling accurate claim submissions, detailed review entries, evidence uploads, and communication inputs aligned with backend API contracts.
- Enhanced UI performance through optimized component rendering, efficient state management, incremental data loading, and error-handling patterns that improved responsiveness across complex multi-view interactions.
- Collaborated with backend, QA, and business teams to refine user flows, adjust UI behaviors, and improve data presentation, ensuring all screens align with claim escalation rules and operational expectations defined by BNSF leadership.

Cloud & DevOps Responsibilities

- Containerized backend and frontend services using Docker and deployed them on Kubernetes clusters with well-defined manifests, health checks, secrets management, and environment configurations enabling scalable, stable, and predictable runtime behavior.
- Utilized **AWS EC2, S3, RDS, CloudWatch, and Lambda** to support compute, storage, monitoring, background processing, event-driven automation, and secure document handling workflows central to the claim resolution process.
- Implemented **CI/CD** pipelines using **Jenkins and GitHub** Actions to automate builds, run static checks, execute test suites, and deploy services across development, staging, and production environments with consistent quality controls.

- Integrated structured logging, error monitoring, and alerting mechanisms using **CloudWatch** and application-level logging frameworks to ensure visibility into system behavior, workflow anomalies, and operational issues.
- Managed **Kubernetes** deployments by configuring autoscaling, load balancing, container orchestration logic, and rollout strategies, ensuring resilient services capable of supporting claim processing workloads without downtime.

Environment: Python, Django, FastAPI, Django REST Framework, React.js, Redux, Hooks, JavaScript (ES6+), HTML5, CSS3, Bootstrap, PostgreSQL, SQLAlchemy, ORM, AWS (EC2, RDS, S3, Lambda, CloudWatch, IAM), Docker, Kubernetes, Jenkins, GitHub Actions, Terraform, Ansible, Kafka, RabbitMQ, Celery, Postman, Swagger/OpenAPI, JIRA, Git, Bitbucket, Agile/Scrum.

Software Engineer **Oct 2021 - Dec 2023**

Sysco, United States

Project Description: Worked on modernizing the Sysco Foods enterprise application used for browsing and managing food products across various categories. The initiative involved transforming a legacy monolithic system into a scalable microservices-based architecture using Python, Django, FastAPI, and PostgreSQL. Responsibilities included developing RESTful APIs, enhancing front-end functionality with React, integrating Swagger for API documentation, and deploying services on AWS. The focus was on improving performance, modularity, and overall user experience across the application.

Backend Responsibilities

- Built backend microservices using Python, Django, and FastAPI as part of Sysco's modernization effort to replace legacy monolithic modules with scalable, decoupled services that improved maintainability and enabled more efficient development workflows.
- Designed and implemented RESTful APIs to manage product categories, metadata, pricing structures, and catalog details while ensuring strong request validation, layered exception handling, and consistent API schema alignment across services.
- Developed PostgreSQL data models and optimized ORM queries to support high-volume product retrieval operations, filtering logic, and category relationship mappings essential for enterprise food catalog management.
- Refactored legacy backend logic into modular service components with structured serialization, asynchronous data processing, and improved data-access patterns that enhanced reliability and clarity across backend operations.
- Integrated Swagger and API Inspector into backend workflows to provide interactive API documentation, streamlined endpoint testing, and improved collaboration between backend, frontend, and QA teams.

Frontend Responsibilities

- Built React.js components using Hooks and Redux to support dynamic product browsing, category filtering, product detail views, and search-driven catalog interactions with smooth rendering and intuitive navigation patterns.
- Developed reusable frontend modules including layout templates, filtering widgets, form controls, modals, and interactive UI blocks designed to improve usability and consistency across catalog browsing screens.
- Implemented state synchronization techniques with Redux to support robust data-fetch flows, error-handling logic, loading indicators, and real-time UI updates based on backend microservices responses.
- Enhanced frontend performance by optimizing component rendering, reducing unnecessary re-renders, implementing lazy loading strategies, and improving request aggregation for complex catalog views.

- Partnered with UI/UX and backend engineers to align API payloads with user-interface needs, resolve integration challenges, and ensure seamless communication between frontend components and distributed microservices.

Cloud & DevOps Responsibilities

- Deployed Python-based microservices onto AWS using EC2, ECS, RDS, S3, and CloudWatch with well-defined configuration patterns, IAM permissions, network setups, and monitoring pipelines supporting secure and reliable cloud operations.
- Used Docker to containerize backend and frontend services and orchestrated deployments with Kubernetes to achieve scalable, fault-tolerant, and consistent runtime environments across development and production systems.
- Built CI/CD pipelines using Jenkins and GitHub Actions to automate code builds, run test suites, manage environment variables, and deploy application updates with controlled branching workflows and deployment strategies.
- Configured CloudWatch dashboards, structured logging, and performance metrics pipelines to detect anomalies, analyze system health, and maintain high operational visibility for product catalog services.
- Managed environment provisioning, service configuration, caching strategies, and deployment automation to ensure optimized performance and reliability throughout cloud-hosted microservices architecture.

Environment: Python, Django, FastAPI, Django REST Framework, React.js, Redux, React Hooks, JavaScript (ES6+), PostgreSQL, SQLAlchemy, Database Migrations, AWS (EC2, ECS, RDS, S3, CloudWatch, IAM), Docker, Kubernetes, Jenkins, GitHub Actions, Swagger / OpenAPI, API Inspector, JSON/YAML, Pandas, Logging Frameworks, Postman, JIRA.

Software Developer **NOV 2019 - Sep 2021**

Independent Bank, United States

Project Description: Worked on developing and enhancing a secure banking operations platform used by internal teams to manage customer information, account activities, transaction workflows, and operational requests. The project involved modernizing legacy components by building microservices using Python, Django, and FastAPI, improving API performance, and creating React-based dashboards for internal banking users. Key responsibilities included designing REST APIs, implementing secure authentication flows, optimizing database operations for financial data, and containerizing services for scalable deployments. The system supported improved accuracy, faster processing, and streamlined day-to-day banking operations.

Backend Responsibilities

- Built microservices using Python, Django, and FastAPI to support customer account workflows, transaction processing, operational requests, and secure data-management modules essential for internal banking operations.
- Developed REST APIs with structured schemas, layered validation, authentication, authorization, and workflow-driven serialization rules enabling safe interactions across internal banking systems.
- Modeled customer, account, transaction, and workflow data structures in PostgreSQL/MySQL with optimized ORM interactions, well-organized relationships, and reliable storage mechanisms supporting regulated environments.
- Enhanced backend reliability by refactoring legacy utilities, optimizing database interactions, applying caching strategies, and improving resource-handling routines across high-throughput banking processes.
- Implemented secure backend patterns such as audit trails, controlled data mutation flows, request-throttling rules, and strict access boundaries aligned with financial institution policies.

Frontend Responsibilities

- Built React.js dashboards and workflow screens using Hooks and Redux to support customer information lookup, account activity monitoring, operational request processing, and approval flows.
- Created reusable UI components including search bars, input controls, data tables, modals, and step-driven workflow containers that improved consistency and usability across internal team interfaces.
- Integrated frontend components with backend APIs using structured data-fetch logic, error-handling layers, loading states, and validation rules aligned with banking workflows.
- Improved the responsiveness and clarity of the user interface by enhancing navigation flows, refining component rendering strategies, and implementing real-time UI updates for critical banking actions.
- Collaborated closely with product analysts, backend engineers, and QA teams to refine screen designs, adjust interface behaviors, and ensure alignment between the UI and complex banking business rules.

Cloud & DevOps Responsibilities

- Containerized backend and frontend services using Docker and deployed them on Kubernetes clusters with scaling rules, health probes, secrets management, and well-defined service configurations supporting stable banking operations.
- Used Terraform and Ansible for infrastructure provisioning, resource orchestration, environment configuration, and controlled deployment setups across multiple banking environments.
- Built automated CI/CD pipelines using GitLab CI, GitHub Actions, and Jenkins to perform code builds, static checks, test execution, and continuous deployment with secure and audited release workflows.
- Implemented structured monitoring, logging, and alerting using CloudWatch, ELK-style tooling, and internal dashboards to maintain operational visibility across all banking microservices.
- Managed environment-level configurations, network policies, cloud resource permissions, and deployment workflows to ensure secure and compliant operations consistent with financial industry standards.

Environment: Python, Django, Flask, FastAPI, Django REST Framework, React.js, Redux, React Hooks, PostgreSQL, MySQL, MongoDB, SQLite, SQLAlchemy, ORM, AWS (EC2, Lambda, S3, CloudWatch, RDS, Secrets Manager), Docker, Kubernetes, GitLab CI, GitHub Actions, Jenkins, Terraform, Ansible, Kafka, RabbitMQ, Celery, Pandas, Logging Frameworks, Git, Bitbucket, JIRA, Confluence, Selenium, PyTest, Postman.

Software Developer **Aug 2017 - Oct 2019**

Innovaccer, India

Project Description: Worked on enhancing and supporting an Electronic Health Records (EHR) platform used by multiple US healthcare hospitals to manage patient demographics, encounters, clinical notes, medication lists, and provider workflows. The project involved modernizing legacy modules by building scalable backend services using Python, Django, and FastAPI, and improving clinical usability with React-based interfaces. Responsibilities included developing RESTful APIs for patient and provider data, implementing secure access layers, optimizing data retrieval for clinical workflows, and ensuring compliance with healthcare standards. The goal was to improve system performance, maintain data accuracy, and streamline the end-to-end clinical documentation experience for hospital staff.

Backend Responsibilities

- Developed backend services using Python, Django, and FastAPI to support EHR modules such as patient demographics, encounters, notes, provider workflows, and clinical document handling with secure validations and audit-driven logic.

- Created REST APIs enabling healthcare staff to retrieve, update, and manage patient records with structured request/response schemas, robust authorization controls, and detailed error-handling workflows.
- Designed database schemas in PostgreSQL/MySQL to manage patient charts, encounter histories, provider data, clinical notes, and workflow activities while optimizing queries to support high-volume clinical operations.
- Integrated secure authentication flows using JWT and OAuth2 to protect sensitive medical information and enforce controlled access aligned with healthcare compliance expectations.
- Enhanced backend performance by optimizing ORM queries, implementing caching strategies, improving serialization processes, and restructuring backend modules for better maintainability and clarity.

Frontend Responsibilities

- Built React.js clinical interfaces using Hooks, Redux, and reusable UI components for patient charts, provider dashboards, visit summaries, encounter timelines, and documentation workflows.
- Developed structured UI patterns including reusable forms, clinical data views, workflow panels, filtering components, and search-driven interactions aligned with healthcare requirements.
- Improved UI responsiveness and clarity by optimizing component rendering behavior, implementing incremental data loading, and refining navigation across complex multi-screen clinical applications.
- Ensured seamless integration between the frontend and backend APIs by designing consistent data-fetch logic, structured error-handling, and clear visual feedback mechanisms for clinicians.
- Worked closely with product teams and clinical analysts to ensure UI flows accurately reflected real-world clinical operations and supported usability improvements across EHR modules.

Cloud & DevOps Responsibilities

- Containerized EHR backend and frontend modules using Docker and orchestrated deployments on Kubernetes clusters to provide scalable, reliable, and consistent infrastructure for hospital environments.
- Leveraged AWS services such as EC2, RDS, S3, CloudWatch, Lambda, and IAM to support secure data storage, document handling, event-driven tasks, audit logging, and end-to-end monitoring across clinical applications.
- Created CI/CD pipelines using Jenkins and GitHub Actions to automate code builds, run tests, publish artifacts, and deploy updates across multiple healthcare environments.
- Implemented structured logging, error monitoring, and diagnostic workflows to maintain operational visibility and ensure reliable behavior during peak clinical usage times.
- Managed infrastructure configurations, deployment scripts, environment variables, and cluster-level parameters ensuring compliance with healthcare data-handling standards.

Environment: Python, Django, FastAPI, Django REST Framework, React.js, Redux, Material UI, PostgreSQL, MySQL, SQLite, ORM, Migration Tools, AWS (EC2, RDS, S3, Lambda, CloudWatch, IAM, SNS, SQS), Docker, Kubernetes, Jenkins, GitHub Actions, Swagger/OpenAPI, JWT, OAuth2, Pandas, Logging Frameworks, Postman, JIRA, Confluence, Agile/Scrum.

EDUCATION

Masters in information systems
Central Michigan University, USA

Bachelors in computer science and engineering
Narayana Engineering College, India