PIYUSH SATTI

+1 (514) 451-1479 Montreal, Quebec, Canada piyushsatti@gmail.com
linkedin.com/in/piyush-satti github.com/piyushsatti

OBJECTIVE

CS candidate and published researcher who writes clean code, takes initiative and ownership, and uses AI to amplify impact. Backend: Java/Python/Node (Express); APIs: REST/GraphQL; Data: Redis/PostgreSQL/MongoDB.

SKILLS

Languages	Java, Python, JavaScript/TypeScript, C/C++, SQL, HTML/CSS, MATLAB
Back-end	FastAPI, Flask, Spring Boot, Express.js/Node.js, REST, GraphQL, OpenAPI
Data	PostgreSQL, MongoDB, Redis, SQLite
Cloud	Docker, DigitalOcean, Vercel, GitHub Actions
Tools & Testing	Git, GitHub, Postman, Linux, Jira/Trello, Figma, Pytest, JUnit, Selenium
Practices	Domain-Driven Design, Event-driven Architecture, TDD, CI/CD, MVC & GoF patterns

EDUCATION

Concordia University (GPA: 3.62/4) Montreal, Quebec, Canada
Master of Science, Applied Computer Science *Sept. 2023 – Aug. 2025*

Thapar Institute of Engineering and Technology (GPA: 9/10) Patiala, Punjab, India
Bachelor of Engineering, Electronics and Computer Engineering *Jun. 2017 – Jun. 2021*

EXPERIENCE

Teaching Assistant Montreal, Quebec, Canada
Concordia University *Jan. 2025 – Apr. 2025*

- Programmer on Duty (Java) for Object-Oriented Programming II with more than 400 students. Guided students to understand OOP concepts and their implementation. Helped with course projects and program debugging.
- Conducted classes for roughly 20 students each week, and a revision lecture for 40 students covering the following topics: File I/O, Polymorphism, Recursion, Exception Handling, Abstract Classes and Interfaces, Inheritance.

Research Assistant Patiala, Punjab, India
Thapar Institute of Engineering and Technology *Aug. 2019 – Apr. 2023*

- Researched algorithm-based techniques for restoring corrupted images. Published 4 academic papers and achieved state-of-the-art performance. Implemented 40 cutting edge research papers in the process.
- Achieved exceptional improvement in the signal-to-noise ratio, resulting in the several papers being published in peer-reviewed journals, including the esteemed IEEE:SPL *(70+ citations).* DOI: **10.1109/LSP.2020.3016868.**

PROJECTS

GameOps Suite: RESTful API + ETL Pipeline piyushsatti/nonagon
Python, FastAPI, Discord API, Redis, MongoDB, Docker *Mar. 2025 – Present*

- Designed and now maintaining an end-to-end telemetry and sign-up platform for *500 Indian Board Gamers* - dashboard for serving player statistics, logic for managing game sign-ups, and Discord interface for in-text AI support.
- Developed an ETL pipeline to convert discord text-based data into MongoDB documents for persistence, inference, and downstream business logic. The pipeline ingests data via Discord's event-driven APIs. Serves it via the API.
- Created RESTful FastAPI endpoints with authentication and role-based authorization for CRUD operations. A dashboard for serving player statistics and displaying real-time game announcements.
- Deployed using containers (API + ETL) over DigitalOcean droplets. The project uses the principles of domain-driven design to keep API logic and the ETL pipeline maintainable. Delivered 200 ms end-to-end latency for typical actions.

Risk-Emulated: Spring Boot + GraphQL piyushsatti/risk-emulated
Java, PostgreSQL, DDD, GoF Patterns, Docker *Aug. 2024 – Present*

- Built a Spring Boot **GraphQL** backend for a playable *Risk* clone with clean Domain-Driven Design: Entities (Game, Player, Territory, Continent, Card, MoveLog), GraphQL types (input/output), **resolvers** for map & turn actions, and Services enforcing rules.
- Designed a graph-based map builder/validator that guarantees connected, legal maps and continent bonuses; exposed a typed **schema** with **queries/mutations** (*createGame*, *reinforce*, *attack*, *fortify*, *gameStatus*).
- Implemented a phase/state engine and rules using GoF patterns (*State* for phases, *Strategy* for combat/reinforcement policies), enabling pluggable variants and targeted test scenarios.
- Dockerized app; GitHub Actions for build/test/coverage; style gates (Checkstyle/Spotless); structured audit logging (MoveLog) for reproducible turns and troubleshooting.