Edwin De Los Santos

Software Engineer

Santo Domingo, Dominican Republic

edwin2delossantos@gmail.com
https://www.linkedin.com/in/edwin-de-los-santos-06b7bab5

Innovative programmer with 7+ years of experience working as a computer programmer. Capable of working with a variety of technology and software solutions, and managing databases. Valuable team member who has experience diagnosing problems and developing solutions.

EXPERIENCE

Bairesdev (*Fabric, Ebay, HealthBridge***) Software Engineer**— *May, 2021 - Present*

- Developed a microservices-based web application using Flask, a lightweight Python web framework.
- Integrated OpenAI's Language Models (LLMs) such as GPT-3 into the backend of the application to enhance natural language processing capabilities.
- Utilized OpenAI's API to interact with the language model for tasks such as text generation, text completion, and question answering.
- Implemented custom wrappers or helper functions to simplify interactions with the OpenAI API and handle authentication and rate limiting.
- Implemented conversational interfaces or chatbots within the application using OpenAI's language model.
- Utilized Flask's routing system to define endpoints for various functionalities.
- Integrated Flask extensions like Flask-Restful for building RESTful APIs and Flask-SQLAlchemy for ORM support with databases.
- Containerized Flask microservices using Docker for consistent deployment environments.
- Defined Dockerfiles to specify dependencies and configurations required for each microservice.
- Utilized Docker Compose to manage multi-container Flask applications for local development and testing.
- Configured Jenkins, a popular CI/CD tool, to automate the build, test, and deployment processes.
- Created Jenkins pipelines using declarative syntax to define stages such as build, test, and deploy.
- Integrated Jenkins with version control systems like Bitbucket to trigger builds automatically upon code commits.
- Implemented pytest, a popular testing framework for Python, to write unit tests and integration tests for Flask applications.
- Created test suites to verify the functionality and behavior of individual endpoints and services.
- Integrated pytest with Bitbucket pipelines to execute tests automatically upon code changes and pull requests.
- Implemented logging within Flask applications to record application events and errors.
- Utilized AWS CloudWatch or ELK (Elasticsearch, Logstash, and Kibana) stack for centralized logging and monitoring of deployed microservices.
- Set up alerts and dashboards to track application performance metrics and respond to incidents promptly.
- Developed frontend components using React, a popular JavaScript library for building user interfaces.
- Leveraged React hooks API to manage stateful logic and side effects within functional components.
- Integrated frontend components with backend microservices using RESTful APIs or GraphQL endpoints.
- Implemented custom hooks to encapsulate API calls and manage data fetching, caching, and error handling.
- Implemented global state management using React Context API for sharing state across multiple components in the component tree.
- Created context providers and consumers to encapsulate and access shared state and functionality throughout the application.
- Wrote unit tests and integration tests for React components using React Testing Library, a lightweight and user-centric testing library.

Tech Stack : Python - Jenkins - Bitbucket - React - Marko - LLM - Flask - Pytest - Playwright - Google Cloud

AlticeDO Mobile Developer — *June, 2020 - May 2021*

- Implemented the new features within the Ionic application using Angular and Ionic Framework.d, enabling seamless communication and data transfer between the front-end and back-end systems.
- Used Ionic components and Angular services to build interactive and responsive user interfaces.
- Leverage Ionic Native plugins to access device-specific features such as camera, geolocation, or push notifications.

- Created unit tests and end-to-end tests for the new features using testing frameworks like Jasmine and Protractor.
- Performed manual testing on multiple devices and platforms to ensure compatibility and consistency.
- Gathered feedback from beta testers or stakeholders to validate the functionality and usability of the new features.
- Implemented data synchronization and authentication mechanisms to securely communicate with the backend.
- Optimized the performance of the Ionic application by minimizing network requests, optimizing images, and lazy loading modules.
- Implemented caching strategies to improve responsiveness and reduce loading times for frequently accessed data.
- Implemented localization and internationalization features to support multiple languages and regions.
- Used ngx-translate to manage translations and provide localized content based on user preferences.
- Generated platform-specific builds for iOS and Android using Capacitor.
- Submitted the app to Apple App Store and Google Play Store, following their respective guidelines and submission processes.
- Identified new features and improvements to be made to the Java backend based on system requirements or business needs.
- Implemented the new features using Java programming language and relevant frameworks such as Spring Boot.

Tech Stack : Ionic - Java - TypeScript Jasmine Capacitor Springboot

APAP Fullstack Developer — *April, 2019 - May 2020*

- Designed and implemented scalable web applications using React.js for frontend and Django for backend, accommodating growing user bases and increasing traffic demands.
- Demonstrated proficiency in both frontend and backend development, leveraging React for building dynamic user interfaces and Django for robust server-side logic and data management.
- Enhanced the Django admin interface to empower users with the ability to configure application parameters directly from the admin dashboard.
- Implemented custom admin views and forms using Django's admin customization features to provide intuitive controls for users.
- Enabled users to adjust various application settings and parameters through the Django admin interface, enhancing flexibility and user control.
- Implemented secure authentication and authorization mechanisms to ensure that only authorized users can access and modify configuration settings.
- Optimized frontend performance by implementing efficient React components, minimizing render times, and optimizing network requests.
- Utilized performance monitoring tools and techniques to identify and address performance bottlenecks in the frontend codebase.
- Designed and developed RESTful APIs using Django REST Framework to facilitate communication between frontend React components and backend Django application.
- Implemented CRUD (Create, Read, Update, Delete) operations and authentication mechanisms to ensure data integrity and security.
- Designed database schemas and models using Django ORM, ensuring data consistency and integrity.
- Utilized database indexing, caching strategies, and query optimization techniques to improve database performance and reduce latency.
- Wrote comprehensive unit tests for both frontend React components and backend Django views, ensuring code reliability and maintainability.
- Automated testing workflows using tools like Jest, Enzyme, and Django's built-in testing framework to streamline the testing process and catch bugs early.
- Implemented continuous integration and continuous deployment (CI/CD) pipelines using tools like Jenkins, GitLab CI/CD, or GitHub Actions.
- Automated build, test, and deployment processes to achieve faster release cycles and ensure code quality and stability.
- Collaborated closely with cross-functional teams including designers, product managers, and fellow developers to deliver high-quality features and meet project deadlines.
- Participated in code reviews, knowledge sharing sessions, and agile ceremonies to foster collaboration and

continuous improvement within the development team.
- Actively collected and incorporated user feedback to prioritize feature development and address user pain points effectively.
- Iteratively improved application usability and functionality based on user input and analytics data.
- Maintained comprehensive documentation for frontend and backend codebases, including architectural decisions, API specifications, and usage guidelines.
- Conducted knowledge sharing sessions and contributed to internal developer resources to promote learning and skill development within the team.

Tech Stack : Python - Javascript - Django - Reactjs - RTL

Octagono Backend Developer — *April, 2019 - May 2020*

- Integrated the Django insurance store with an Invoicing system using Odoo's Remote Procedure Call (RPC) API.
- Established secure connections to the Odoo server and authenticated API requests to access Invoicing system functionalities programmatically.
- Implemented bidirectional data synchronization between the Django insurance store and the Odoo Invoicing system.
- Used Odoo RPC API methods to retrieve, update, and synchronize insurance-related data such as customer information, policies, premiums, and claims.
- Leveraged Odoo's Invoicing API endpoints to automate billing processes for insurance premiums, policy renewals, and claim settlements.
- Integrated with Odoo's invoicing workflows to generate and send invoices, track payment status, and reconcile accounts receivable.
- Created custom API endpoints in the Django backend to expose specific functionalities of the Odoo Invoicing system.
- Implemented endpoints to handle operations such as creating invoices, updating payment status, querying customer balances, and retrieving invoice details.
- Developed RESTful API endpoints in the Django backend to facilitate communication between frontend applications and the integrated systems.
- Designed API endpoints following best practices such as resource naming conventions, HTTP method semantics, and error handling mechanisms.
- Implemented secure authentication mechanisms for accessing Odoo RPC API and Django RESTful API endpoints.Implemented robust error handling mechanisms to gracefully handle exceptions and failures during API communication.
- Utilized retry strategies, exponential backoff, and circuit breaker patterns to improve fault tolerance and resilience in the integration layer.
- Documented API endpoints, request/response payloads, and usage guidelines for frontend developers consuming the Django backend APIs.
- Optimized API performance by minimizing latency, reducing network overhead, and implementing caching strategies where applicable.
- Collaborated with stakeholders including business analysts, product owners, and frontend developers to gather requirements and prioritize integration tasks.
- Communicated effectively with cross-functional teams to align on integration goals, timelines, and deliverables.

Tech Stack : Python - Odoo - Django - RPC - REST

SKILLS

Programming Languages: Advanced: JavaScript, Python, Java.
Frameworks Tools: Advanced: React, Angular, Ionic, Django, Odoo, React Native.
Data Base: Advanced: Firebase, Postgresql; Advanced: MySql.
Tools: Jenkins, AWS SES, AWS SNS, AWS S3, Azure Datalake, Azure 0auth, Openshift, LLM.
Agile: Kanban, Scrum.
Languages: Advanced: English; Native: Spanish.

EDUCATION

UNAPEC – Santo Domingo. *Software Engineering, May 2021*