

Design and Implementation of Jelly fish AI: An Automated Gen AI Development Platform for Full-Stack Software Engineering

Aniket Ubale, Pankaj Mhaske, Akshay Sange, Neha Varma , Prof. Vijayalaxmi Tadkal

Bachelor's Degree in Computer Engineering AIML & Bharat College of Engineering

Abstract - Modern program development demands builders to manage diversified tasks such as systematize, experiment, deployment, and method monitoring. These processes frequently demand knowledge of various programming foundations and cloud sciences, making development complex and behind. Jellyfish AI is an automated happening manifesto designed to facilitate full-stack spreadsheet design using Generative Artificial Intelligence. The system admits users to name request requirements in human language, after that the podium automatically create project structures, use law, and deployment configurations. The program still integrates containerization and orchestration electronics to ensure trustworthy and ascendable deployment of requests. In addition, Jellyfish AI provides a listening instrument panel that helps developers path application act and scheme resources. By joining AI-based law production with electronic deployment and administration finishes, the proposed order aims to reduce growth work and improve builder productivity. This policy displays how brilliant development atmospheres can organize software metallurgy workflows and support faster application growth.

Key Words: Generative Artificial Intelligence, Automated Code Generation, Full-Stack Development, Cloud Deployment, Containerization, Kubernetes Orchestration, DevOps Automation, Intelligent Development Platform

1.INTRODUCTION

Modern operating system development has progressed briskly with the progress of machine intelligence, cloud computing, and DevOps practices. Developers contemporary proper to manage diversified stages of the software happening lifecycle, containing requirement study, systematize, testing, arrangement, and listening. Managing these processes manually often demands expertise in various finishes and technologies, that can create development complex and behind, exceptionally for beginners and narrow development crews. Generative Artificial Intelligence has currently emerged as a strong electronics that can assist developers in miscellaneous programming tasks. These models are fit understanding human language instructions and create significant outputs such as law particles, documentation, and organized project components. By handling these efficiencies, developers can reduce the growth process by describing their use necessities in simple word while the system without thinking produce the required law form. Another important happening in new software planning is the adoption of containerization and cloud-located arrangement technologies. Containers authorize requests to run in unique surroundings with all unavoidable dependencies contained, guaranteeing consistent conduct across various systems. Container written music manifestos help in managing diversified containers, automating arrangement, measuring applications, and listening whole resources capably. To simplify the complicatedness complicated in modern spreadsheet growth, this research introduces Jellyfish AI, an inventive principle designed to mechanize the development and arrangement of adequate-stack applications. The projected order integrates generative machine intelligence accompanying containerization and orchestration electronics to assist developers during the whole of the growth lifecycle. Developers can provide request necessities using human language, afterwards which bureaucracy automatically create project forms, code modules, and arrangement configurations.

1.1 Project Overview

Jellyfish AI is an bright development policy devised to simplify the process of construction and deploying entire- stack applications. Modern operating system happening often demands builders to work with diversified forms for coding, experiment, arrangement, and monitoring. This process maybe complex and behind, especially for novices the one may not have knowledge accompanying cloud technologies and canister-located environments. The projected Jellyfish AI whole uses Generative Artificial Intelligence to assist developers by inevitably create project structures and elementary use code established consumer requirements. Developers can interpret their use in simple vocabulary, and the system create the necessary components for happening. The manifesto also supports containerization and cloud arrangement electronics to ensure that uses run usually across different surroundings. By joining automated rule era with arrangement forms and monitoring visage, Jellyfish AI aims to shorten the development plan and boost productivity for builders.

1.2 System Objectives

The main objective of the Jellyfish AI terrace is to cut down the growth and deployment of thorough-stack spreadsheet applications by merging Generative Artificial Intelligence accompanying modern cloud electronics. Traditional incident workflows require planners to manually design use structures, print abundant amounts of code, and construct arrangement environments. These processes maybe behind and may demand knowledge in multiple forms and foundations. Jellyfish AI aims to automate these tasks by produce application law and project forms based on consumer necessities. The system also supports canister- located deployment and determines listening features that help builders path application conduct. By joining artificial intelligence accompanying happening and deployment finishes, the policy seeks to develop growth efficiency and humble the complicatedness involved in up-to-date program engineering.

2. Review of Literature

Recent progresses in Artificial Intelligence and software metallurgy have considerably improved the habit new applications are grown and redistributed. Many researchers have concentrated on integrating smart orders with program happening tools in consideration of defeat manual effort and increase output .Several studies highlight the use of AI- helped prioritize tools that help builders produce code as a matter of usual practice based on consumer recommendation or predefined templates. These tools resolve abundant datasets of programming rule and support suggestions to planners while writing uses. Such approaches correct development speed and weaken systematize errors. Research in cloud estimating has still contributed tomechanized application arrangement utilizing containerization technologies. Containers admit uses to run in private surroundings with agreeing configurations, which reduces arrangement across different policies. Technologies in the way that container musical adaptation orders enable planners to manage diversified buckets efficiently and guarantee extreme system chance. Other studies devote effort to something DevOps practices that integrate happening and deployment workflows. DevOps finishes mechanize processes such as construction, experiment, and deploying applications, that reduces the time necessary to release operating system updates. Although these sciences have enhanced software growth processes, many existent systems still demand developers to manually mix various tools for systematize arrangement, and monitoring. This increases complicatedness and demands additional mechanics expertise .The Jellyfish AI floor addresses these restraints by combining fruitful machine intelligence, automated arrangement means, and monitoring finishes into a single united surroundings. This integrated approach clarifies the brimming-stack development process and helps planners build and redistribute applications more capably.

2.1 Existing Systems

Numerous spreadsheet growth tools once lie that assist planners in writing law, directing requests, and deploying software to cloud policies. However, most of these orders devote effort to something only one some the happening lifecycle alternatively providing a fully joined answer. Some of the usually used podiums are:

- GitHub Copilot & Code Whisperer – These tools provide AI-based code suggestions and help developers write code faster. However, they mainly assist with small code snippets and do not generate complete full-stack project structures or handle deployment automatically.
- Low-Code Platforms (Bubble, OutSystems) – These platforms allow users to build applications with minimal coding. Although they simplify development, they often limit customization and may not support complex application architectures.
- Docker & Kubernetes Platforms – These technologies enable containerization and orchestration of applications. While they provide powerful deployment capabilities, they require technical knowledge to configure infrastructure.
- CI/CD Tools (Jenkins, GitHub Actions) – These tools automate software build and deployment processes, but developers still need to configure workflows and integrations manually.
- Cloud Development Platforms – Various cloud platforms offer development and hosting services, but they often require separate configurations for coding, deployment, and monitoring.

2.2 Literature Survey of Similar Ideas

- Several research studies have surveyed the use of Artificial Intelligence to assist program development processes. AI-stimulate systematize assistants and computerized incident planks have gained meaningful consideration for improving planner output and lowering manual coding work.
- GitHub Copilot – This finish uses machine learning models prepared on abundant law repositories to create law plans for developers. It helps in manuscript functions and accomplishing code blocks certainly but does not create a complete thorough-stack application form.
- Amazon Code Whisperer – It specifies intelligent rule pieces of advice all along development and helps programmers compose rule efficiently. However, it chiefly focuses on rule creation within the growth atmosphere and does not handle automated arrangement or foundation administration.
- Low-Code and No-Code Platforms – Platforms such as Bubble and OutSystems admit consumers to build applications utilizing able to be seen with eyes interfaces accompanying minimal systematize. While these principles simplify incident, they frequently limit customization and adaptability required for complex spreadsheet plans.
- Containerization Technologies – Tools such as Docker authorize planners to bundle applications and their reliances into boxes, guaranteeing consistent killing across various environments. Container musical arrangement planks help control multiple bags in delivered systems.
- DevOps and CI/CD Tools – Continuous unification and arrangement finishes automate the process of construction, experiment, and deploying applications. These finishes help program delivery speed but still demand manual arrangement and integration accompanying happening atmospheres.
- Although these technologies help various stages of the development lifecycle, most of ruling class manage as free tools. Developers frequently need to merge multiplepodiums for systematize, arrangement, and monitoring. The projected Jellyfish AI whole aims to combine these skills into a distinct brilliant platform that reduces the whole operating system development plan.

3. Proposed System

The projected system presents an joined and brainy platform created to streamline the process of software growth and arrangement. The Jellyfish AI system focuses on automating various stages of the spreadsheet development lifecycle by joining generative machine intelligence, cloud sciences, and containerization tools into a alone terrace .The platform admits builders to generate request buildings and code parts automatically established consumer requirements. By utilizing AI-located development help, bureaucracy helps reduce manual systematize effort and betters output during the growth process. Developers can focus upon designing arrangement use rather than giving overdone time on repetitious coding tasks .In addition to rule production, the proposed arrangement supports computerized deployment utilizing bottle-based sciences. Applications maybe packaged into bags and deployed capably inside cloud environments. The system again specifies monitoring lineaments that help path the performance and rank of redistributed applications .By mixing development, arrangement, and listening functionalities into a single atmosphere, Jellyfish AI shortens the full-stack incident plan and provides a organized approach for building up-to-date requests.

3.1 Analysis / Framework / Algorithm

Jellyfish AI plank is created to mechanize different stages of the operating system growth lifecycle utilizing Generative Artificial Intelligence and cloud sciences. The foundation focuses on simplifying application happening, containerization, arrangement, and listening through an smart and joined system. The following steps detail the occupied foundation of the projected plan:

Step1:Problem Identification

Issue: Modern software incident demands information of diversified finishes to a degree coding foundations, cloud aids, and arrangement policies.

Solution: Develop an AI-stimulate platform that facilitates the process of construction and deploying adequate-stack requests.

Step 2: Requirement Input

Issue: Developers frequently spend time plotting project constructions before offset happening.

Solution: The system admits users to support use necessities, that are therefore processed for one AI model to create the beginning project construction.

Step 3: AI-Based Code Generation

Issue: Writing repetitious code parts can hinder growth.

Solution: Generative AI models without thinking produce code templates and modules established consumer necessities.

Step 4: Containerization

Issue: Applications concede possibility function differently across surroundings on account of reliance conflicts.

Solution: The system bundle uses into containers to guarantee logical killing across manifestos.

Step 5: Orchestration and Deployment

Issue: Managing diversified crates and services maybe complex.

Solution: Container musical adaptation forms survive arrangement, scaling, and help ideas.

Step 6: Monitoring and Logging

Issue: Developers need to monitor request act following in position or time deployment.

Solution: The podium integrates listening finishes to path whole performance, logs, and support custom.

Step 7: User Interface and Interaction

Issue: Complex forms create growth difficult for learners. **Solution:** JellyfishAI supports an smooth-to-use instrument panel that admits users to control projects, deployments, and listening ventures.

Step 8: Scalability and Optimization

Issue: Applications must handle growing workloads capably.

Solution: The system supports scalable arrangement construction to accomplish extreme workloads and claim performance.

Step 9: Security and Access Control

Issue: Security risks concede possibility happen when uses are redistributed in cloud surroundings.

Solution: Authentication and secure approach mechanisms are executed to insulate consumer dossier and method resources.

Step 10: Result Generation

Issue: Developers demand clear observations about plan accomplishment.

Solution: The floor provides ocular instrument panels and reports that help consumers resolve arrangement status and request conduct.

3.2 System Architecture (Challenges in Automated Software Development Platform)

Existing program incident platforms face various challenges in automating the complete lifecycle of use growth, especially when mixing machine intelligence, containerization, and arrangement systems. The following issues climax structural and working limitations in addition to their projected answers:

1. Requirement Understanding and Processing Challenge:

Many development manifestos demand builders to manually define itemized project forms and structure requirements before happening starts, that increases time and complicatedness.

Solution:

The system uses AI-located necessity processing that resolves consumer recommendation and automatically produce the primary project form and development plan.

2. AI-Based Code Generation Challenge:

Developers frequently give a significant amount momentary novel repetitious and boilerplate law all along program development.

Solution:

Integrate Generative AI models that instinctively produce law templates, modules, and basic program makeups established consumer requirements, lowering happening work.

3. Environment Consistency and Dependency Management Challenge:

Applications may properly otherwise across miscellaneous environments on account of reliance conflicts and arrangement differences.

Solution:

Use containerization sciences to a degree Docker to bundle applications in addition to their reliances, guaranteeing consistent killing across happening, experiment, and production atmospheres.

4. Deployment and Orchestration Challenge:

Managing diversified canisters and services in a delivered order can enhance complex and difficult to control.

Solution:

Implement canister musical adaptation methods such as Kubernetes, that mechanize arrangement, scaling, and administration of containerized uses.

5. Monitoring and System Performance Challenge:

Developers frequently lack actual-time visions into use conduct after arrangement.

Solution:

Integrate listening and record tools that path scheme act, application logs, and system exercise, allowance developers fast discover and resolve issues.

6. User Interface and Platform Accessibility Challenge:

Many growth platforms have complex interfaces that form bureaucracy troublesome for beginners to use.

Solution:

Jellyfish AI supports a natural and common dashboard that admits consumers to control projects, generate law, redistribute requests, and monitor performance surely.

7. Scalability and Resource Optimization Challenge:

Applications must handle growing workloads capably without moving accomplishment.

Solution:

The principle uses scalable cloud foundation and box written music to dynamically allocate money established use demand.

8. Security and Access Control Challenge:

Cloud-based incident principles concede possibility face security risks to a degree unofficial approach and data breaches.

Solution:

Implement confirmation devices, secure APIs, and part-based approach control to guarantee secure access to bureaucracy and cover consumer data.

9. Integration of Development Tools Challenge:

Developers frequently need to use diversified forms for coding, arrangement, and listening, that complicates the plan.

Solution:

Jellyfish AI integrates law production, containerization, deployment, and listening finishes into a alone platform, simplifying the complete incident lifecycle.

10. Result Visualization and Reporting Challenge:

Developers demand clear insights into project rank and arrangement results.

Solution:

The system supports visual instrument panels, data reports, and arrangement summaries that help consumers appreciate method performance and project progress efficiently.

3.3 Data Model

The data model for the Jellyfish AI principle follows an Entity-Relationship Model (ERM) to guarantee organized and efficient dossier administration. It consists of:

- **User Profile:** Stores consumer facts, confirmation details, project purchase, and action history inside the policy.
- **Project Data:** Contains project necessities, generated rule modules, project form, and configuration analyses established by the AI structure.
- **Deployment Data:** Stores bucket countenances, deployment configurations, musical arrangement analyses, and environment scenes second hand during use arrangement.
- **System Logs and Monitoring Data:** Maintains use logs, performance versification, mistake reports, and system listening facts to path platform accomplishment and troubleshooting activities.

3.4 Methodology

To guarantee influence and scalability, the Jellyfish AI terrace is developed utilizing a orderly approach. The strategy resides of:

AI Requirement Processing: The system accepts consumer project necessities through an interactive connect and processes ruling class using AI models to inevitably produce an beginning project structure.

Generative AI Code Generation: Pre-prepared fruitful AI models are used to certainly generate law templates, modules, and incident components necessary for construction entire-stack applications.

Containerization Technology: Containerization finishes to a degree Docker are used to package uses in addition to their reliances, ensuring logical killing across development and arrangement atmospheres.

Orchestration and Deployment: Container musical arrangement platforms to a degree Kubernetes control containerized applications, permissive electrical arrangement, scaling, and capital administration.

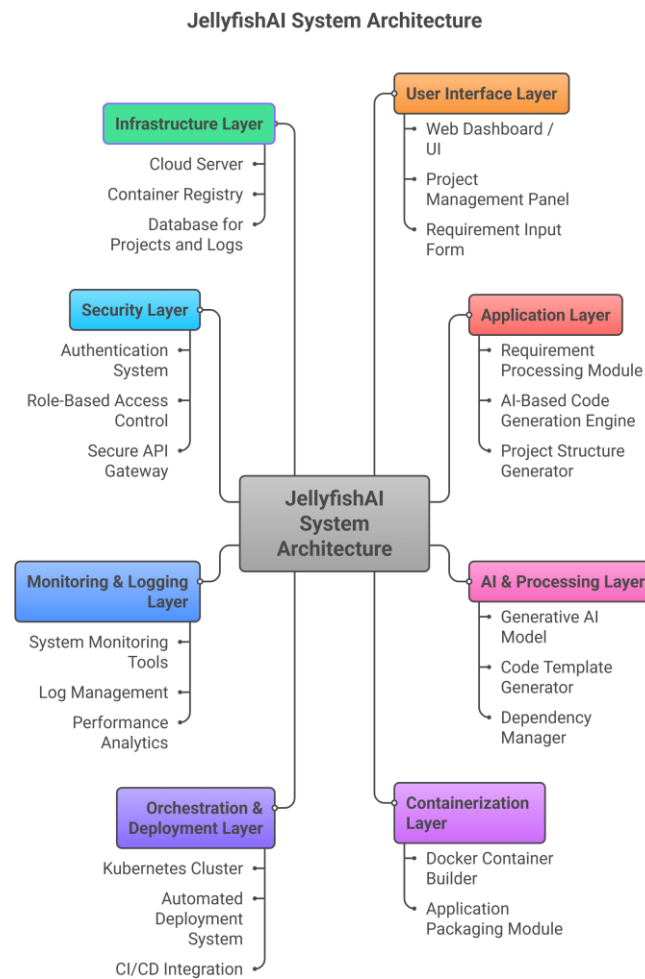


Figure 1.1 System Architecture

• **Monitoring and Logging:** System monitoring forms path use performance, logs, and order action, course developers monitor deployments and fast resolve issues.

• **User Interface and Dashboard:** A plain and intuitive instrument panel admits consumers to manage projects, create rule, deploy uses, and monitor scheme accomplishment efficiently.

• **Performance Optimization:** The program applies effective resource distribution, climbable cloud foundation, and optimized capsule administration to ensure fast refine and trustworthy structure performance.

4.1 Proposed System Result

The achieved Jellyfish AI principle provides an electrical and effective approach to software growth by merging artificial intelligence, containerization, and arrangement electronics into a single manifesto. The system facilitates the process of building and deploying entire-stack requests by as a matter of usual practice generating law, constructing project structures, and directing use deployment.

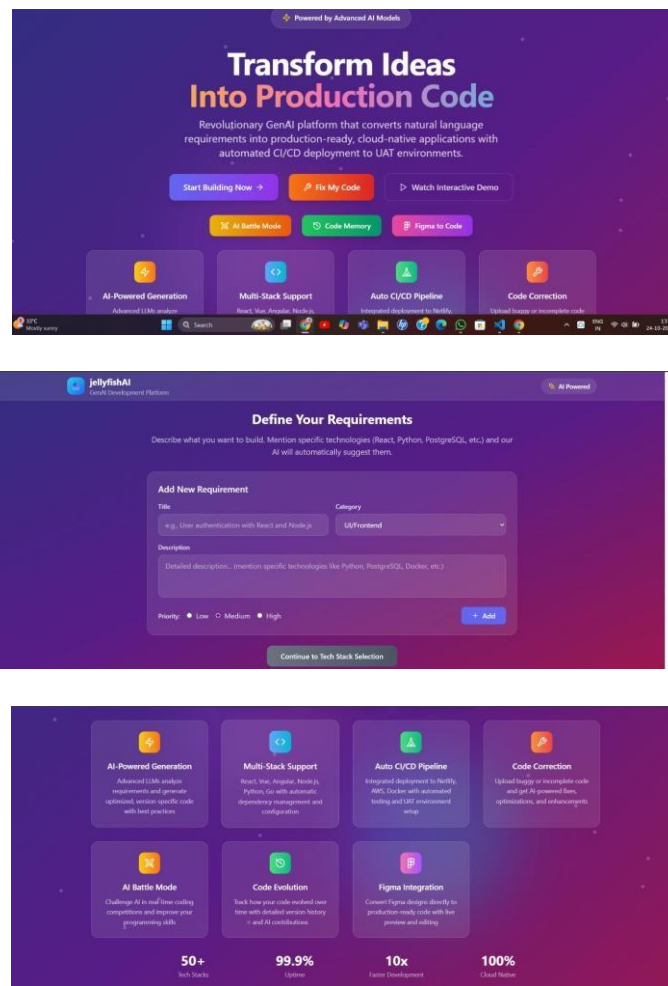


Figure 2 . Snapshots of home and deployment page

5. Conclusion

The Jellyfish AI platform provides an intelligent approach to software development by integrating artificial intelligence, containerization, and automated deployment into a single system. It enables users to generate project structures and code components based on requirements, simplifying the development process and reducing manual effort. By utilizing technologies such as Docker and Kubernetes, the platform ensures scalable deployment and efficient application management. Overall, the system improves productivity and provides a streamlined environment for building and deploying modern applications, with future scope for advanced AI capabilities and improved cloud integration.

References

- [1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, Deep Learning. MIT Press, 2016.
- [2] Martin Fowler, Continuous Integration: Improving Software Quality and Reducing Risk. Addison-Wesley, 2006.
- [3] Docker Inc., "Docker Documentation," Available: <https://docs.docker.com>
- [4] Cloud Native Computing Foundation, "Kubernetes Documentation," Available: <https://kubernetes.io/docs/>
- [5] OpenAI, "Generative AI and Large Language Models for Software Development," Available: <https://openai.com>

- [6] Amazon Web Services, "Cloud Computing Concepts and Deployment Models," Available: <https://aws.amazon.com>
- [7] Gene Kim, Kevin Behr, and George Spafford, The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win. IT Revolution Press, 2013.
- [8] Linux Foundation, "Cloud Native Application Architecture," Available: <https://www.linuxfoundation.org>
- [9] Google Cloud, "Containerized Application Development and Deployment," Available: <https://cloud.google.com>
- [10] IEEE, "Artificial Intelligence for Software Engineering: Trends and Applications," IEEE Research Publications.

Acknowledgment:

We express our gratitude to our project guide Prof. Vijayalakshmi Tadkal, Assistant Professor, Department of Computer Engineering AIML for her valuable suggestions, cooperation, and support in the working of this paper.