The Onboarding Manager

Lucas Kazem, Shuban Sridhar, Jett Morrow, Richmond Southall
(lucaskazem, shubansridhar, jettmorrow, richmonds)

Abstract

The Onboarding Manager is a software solution designed to streamline the onboarding process for new software developers. It addresses common challenges such as setting up complex development environments, navigating scattered documentation, and understanding existing project architectures. The system automates key onboarding tasks, including configuring repositories, sharing project documents, and assisting developers in comprehending codebases. By reducing reliance on team members and standardizing the onboarding process, the Onboarding Manager minimizes productivity losses and accelerates the integration of new developers into ongoing projects. This project adopts an incremental development approach, allowing continuous delivery and refinement of features based on user feedback. The Onboarding Manager aims to improve the onboarding experience for developers, increase team efficiency, and establish a robust, scalable process for integrating new members into software development teams.

Introduction

The onboarding process for software developers is a critical yet challenging phase in any project. New developers often face steep learning curves due to complex development environments, complicated project architectures, and scattered or incomplete documentation. These challenges not only slow down the integration process but also require a lot of time and effort from existing team members, taking them away from their core responsibilities. This inefficient onboarding process can lead to reduced productivity, errors, and frustration for both the new developers and the team as a whole.

The Onboarding Manager project addresses these issues by providing a structured, automated solution to simplify and speed up the onboarding process. The tool automatically configures development environments, sets up necessary repositories, and ensures that all critical project documents are accessible to the new developers. Additionally, it assists in understanding existing codebases through clear documentation and code navigation aids, enabling developers to contribute more quickly and effectively.

The need for a solution like the Onboarding Manager is evident in the modern software development environment, where rapid onboarding is essential to maintain team productivity and meet project deadlines. Studies show that an efficient onboarding process not only reduces time-to-productivity but also increases job satisfaction and retention among new developers. [1] By standardizing and automating key aspects of onboarding, the Onboarding Manager aims to enhance the developer experience, minimize disruptions for existing team members, and foster a more efficient, high-performing development environment.

Motivating Example

Let us picture a large software development company onboarding new engineers for a high-priority project. Usually, this process

involves manual setup of development environments, explanations of codebased and documentation, and extensive reliance on senior team members to assist the new hires. These inefficiencies in the onboarding process can lead to delays and an overall slow ramp-up process, wasting time that can be spent on developing code and meeting deadlines. The Onboarding Manager solves this problem beginning from day one. Upon a new developer's first day of work, they log into the Onboarding Manager. The system automatically configures their environment by cloning repositories, installing dependencies, and setting permissions, all of which have been pre-configured by their team lead. The new developers can access project documentation immediately, eliminating the need to request access manually and search through large repositories for proper documentation. During this process, the Task Progress Tracker presents a clear roadmap to the developers, guiding them through their initial tasks.

On day two, developers can utilize the Automated Documentation Overview to begin understanding the structure of their project. The system provides a high-level overview of key modules and their relationships, eliminating the struggle many new developers share when they feel overwhelmed by scattered documentation and advanced architecture. By day three, the new developer is ready to begin contributing to their project. The Onboarding Manager greatly reduces the reliance on senior developers for guidance, and new developers can quickly begin completing tasks, saving the team valuable time and improving productivity.

Background

For developers, onboarding often involves:

- Development Environment Setup: Configuring local or remote environments to match the project's requirements, including frameworks, dependencies, and database connections.
- Repository Management: Identifying and cloning the correct set of repositories and understanding branching strategies, continuous integration pipelines, and code review processes.
- Documentation Access: Locating and comprehending the necessary documentation, including architectural diagrams, API references, coding standards, and style guidelines.
- Codebase Familiarization: Gaining insights into the existing codebase, its modules, key classes, and patterns, often assisted by automated code exploration tools or integrated development environment (IDE) features.

Related Work

Onboarding plays a critical role in shaping a new hire's experience within a company and setting the tone for their future success. In the fast-paced tech industry, where turnover rates average 13.2% and rise to 21.7% for software engineers, the importance of a strong onboarding process cannot be overstated. When developers leave, it can cost a company up to 250% of their annual salary in recruitment, onboarding, and lost productivity. [2] A well-structured onboarding process not only accelerates a developer's ability to contribute but also fosters a sense of belonging, support, and alignment with the company's culture. Without this foundation, new employees often struggle to integrate, leading to dissatisfaction and increased turnover.

Several tools exist to aid in onboarding and streamlining workflows, such as Garden and Whatfix. Garden focuses on providing developer-friendly tools for automating workflows and testing within a Kubernetes

environment. [3] It streamlines developer operations and improves productivity but does not address the broader onboarding experience for new hires. Garden is primarily focused on optimizing development pipelines rather than guiding new developers through the process of setting up their environments or learning about existing projects. Similarly, Whatfix provides a digital adoption platform that creates interactive walkthroughs to help users understand software applications.[4] While this tool is effective for training and onboarding end users, it does not address the unique needs of software engineers during onboarding. It lacks the ability to integrate with developer tools, automate environment setup, or assist in understanding project-specific codebases and workflows.

The Onboarding Manager fills this gap by focusing on the specific challenges faced by software engineers during onboarding. Unlike general-purpose tools, it provides tailored support for integrating new developers into projects quickly and efficiently. By automating environment setup, sharing project resources, and helping developers understand existing code and documentation, the Onboarding Manager directly addresses the pain points of software engineering onboarding that other tools overlook. This targeted approach ensures that new engineers can begin contributing productively sooner, reducing onboarding time and improving overall team efficiency.

Design, Implementation, and Testing

Requirements Analysis

Functional Requirements:

1. Centralized Access and Permissions Management (High Priority):
   The system must automate the process for requesting and granting access permissions.
2. Automated Development Environment Setup (High Priority):
   The system must configure development environments, including repository cloning and dependency installations, without senior developer intervention.
3. Automated Codebase Documentation Overview (Medium Priority):
   The system must generate a high-level overview of the codebase structure.
4. Integration with Existing Tools and Systems (Medium Priority):
   The system must integrate with project management and communication tools.
5. Task Progress Tracking System (Medium Priority):
   The system must provide a visual dashboard of completed, in-progress, and pending onboarding tasks.

Non-Functional Requirements:

- Usability: Complete setup within 1 hour, scoring at least 75% in usability tests.
- Reliability: Maintain 99% uptime during business hours and recover critical failures within 24 hours.
- Performance: Respond within 2 seconds for user interactions and support up to 50 concurrent sessions with minimal performance degradation.
- Supportability: Provide updates within 24 hours for critical issues and achieve a 95% first-attempt resolution rate.
- Implementation/Constraints: Use Java, ensure at least 80% unit test coverage, and maintain compatibility with existing organizational tools.

Use Cases

1. Requesting Access to Project Resources:
   New developers request project access,

automatically granted based on pre-registered profiles.
2. Automated Environment Setup: A project manager adds a new developer to a project, triggering automatic environment configuration.
3. Adding New Developer Tool: The project manager introduces a new tool, and all developers' environments update automatically.
4. Removing Developer Tool: The project manager removes an unnecessary tool; developers' environments adjust accordingly.
5. Access to Documentation Database: Developers access a documentation database to find relevant project information.

Incremental Development Approach

Increment 1: Centralized Access and Permissions Management

- Automates granting of resource access based on pre-configured profiles.
- Includes a notification system for successful access grants.

Increment 2: Automated Development Environment Setup

- Automates repository cloning, dependency installation, and environment variable configuration.
- Displays a status dashboard for real-time setup progress.

Increment 3: Automated Codebase Documentation Overview

- Parses code to produce structural overviews and relationships.
- Offers search and query features for quick navigation.

Increment 4: Task Progress Tracking System

- Provides a dashboard categorizing tasks as completed, in progress, or pending.
- Sends notifications for deadlines and overdue tasks.

This incremental model allows early delivery of core functionality (permissions, environment setup) and subsequent integration of documentation and tracking features.

Testing Approach

The testing approach for the Onboarding Manager is designed to ensure that the system aligns with the project scope, meets both functional and non-functional requirements, and adheres to the design decisions established. The provided test cases validate critical aspects of the system that aim to improve and automate the developer onboarding process. By covering tasks such as automated repository setup, centralized access management, task progress tracking, automated codebase documentation, and the integration or removal of development tools, these tests confirm that the system's features function as intended and deliver value to end-users.

Functional and Non-Functional Requirements Coverage

The included test cases verify multiple core functional requirements of the system:

- Automated Repository Setup (Test ID: 1) ensures that new developers can have their environments and repositories configured without manual intervention.
- Centralized Access Management (Test ID: 2) validates that developers are granted the necessary permissions to documentation and tools.
- Task Progress Tracking (Test ID: 3) checks that developers can easily

monitor their onboarding progress in real time.

- Automated Codebase Documentation (Test ID: 4) confirms that the system can parse commits and store associated documentation.
- Codebase Documentation Search (Test ID: 5) ensures that developers can filter and retrieve project-specific documentation.
- Adding a New Development Tool (Test ID: 6) and Removing a Development Tool (Test ID: 7) verify that developers can update their development environments seamlessly.
- Role-Specific Onboarding Tasks (Test ID: 9) confirm that the onboarding experience is tailored to the developer's job role.

Additionally, the test cases cover non-functional requirements such as usability and performance by verifying that developers can easily access required resources, monitor their progress, and integrate new tools with minimal effort. For example, Onboarding Feedback Collection (Test ID: 8) ensures that feedback is recorded for continuous improvement, and System Notifications for Onboarding Milestones (Test ID: 10) validate that users and administrators receive timely updates on important onboarding events.

Deploying and Maintaining

Deploying the Onboarding Manager would involve setting up an automated process to build, test, and release the software. This ensures the system is reliable and any updates can be introduced smoothly. A staging environment would be used to test changes before they are released to users, which reduces the risk of introducing errors into the live system. Concepts like continuous integration and continuous delivery (CI/CD) are key here, as they help automate and speed up the deployment process.

Maintaining the system would involve monitoring its performance and fixing any issues that arise. Regular updates would address bugs, improve security, and add new features. This reflects the idea of iterative improvement, a concept often discussed in software engineering, which focuses on making continuous enhancements to meet user needs. By combining proper deployment strategies with ongoing maintenance, the Onboarding Manager would remain reliable, efficient, and effective for developers.

Limitations and Future Extensions

Limitations

Scalability Constraints

Currently, the system is designed and optimized for small to medium-sized teams. Expanding to enterprise-scale usable where thousands of developers may be onboarded simultaneously would require proper testing with a larger scope. For example, high loads may complicate task automation and documentation generation which would need to be accounted for in our system. And under heavy usage, bottlenecks may occur with features such as repository setup and automated documentation parsing. Future enhancements such as database optimization, load balancing, and distributed architectures can help address these challenges.

Integration Complexity

Integrating with third-party tools, such as different IDEs and project management tools,

would require customization for various use cases. Different organizations use a wide variety of tools, some of which may have limited APIs or unique configuration settings which we must account for. Teams using unique tools or workflows may face challenges in adopting the system without additional development. Offering support for commonly used tools and allowing for custom plugins can help reduce this complexity.

Usability Testing and Feedback
The system's user interface and workflows would require border testing across various teams, roles, and organizations in order to properly identify and address potential usability issues. Currently, testing has been limited to small sample groups. Key challenges exist such as UI and UX optimization. The onboarding dashboard and task tracking feature likely require more testing in order for those features to align with the needs of developers with varying technical expertise. Additionally, limited feedback has been collected on system workflows, which may not align with real-world team dynamics. Future iterations should prioritize conducting extensive usability tests to gather feedback that can be applied to real-world usage.

Security and Access control
The system currently includes features to automate access to repositories, tools, and project documentation, but it lacks advanced security features such as encryption and multi-factor authentication which are essential in high-risk, real-world projects. Further iterations must work towards protecting sensitive documentation and metadata from authorized access, and authentication and data encryption standards to secure user data. Robust security features must be incorporated into any related technology to prevent data breaches or compliance risks.

Future Extensions

Advanced Documentation Search and Filtering
The current system includes a feature to support basic searches across a codebase. Expanding this functionality will further enable developers to locate precise documentations. Future features should expand on this by adding support for inline documentation or keywords within specific project directories. Modern technologies such as machine learning and natural language processing could also be incorporated to rank search results by relevance. Visualization tools could also enhance this feature by generating dependency graphs to show relationships between files, modules, and classes.

Customizable Onboarding Workflows
Currently, the system consists of a fixed onboarding workflow that may not align with the specific needs of all teams or roles. Introducing role-based onboarding templates with customizable options will help provide flexibility for diverse teams. For example, project managers can design different onboarding workflows tailored to specific roles, like front-end or back-end developers, where each role has different prioritized tasks. Tasks can also be automatically assigned based on the developer's role or project, and role-specific tools like style guides for front-end developers can be integrated into our system.

Multilingual Documentation Support
Incorporate automated language translation capabilities for project documentation and onboarding materials. This would help international teams by providing developers with resources in their preferred language, enhancing understanding, and lowering the language barrier that can slow down onboarding.

Smart Documentation Summaries
Leverage AI-powered text summarization to provide concise, relevant snapshots of large documents or architecture descriptions. This feature would give new developers just enough context at a glance, helping them quickly identify what's crucial without reading through lengthy manuals.

Onboarding Analytics Dashboard
Provide project managers and HR personnel with a high-level dashboard showing key onboarding metrics—average time to first commit, user satisfaction scores, number of support requests, and common bottlenecks. This data-driven view helps stakeholders continuously refine and improve the onboarding process.

Conclusion

The Onboarding Manager not only addresses common problems in the onboarding process but greatly reduces dependency on senior developers. Now, those experienced team members can focus on their core tasks rather than having to guide new developers through their onboarding processes. By streamlining access, environment setup, and documentation, the Onboarding Manager ensures that new developers can begin contributing to a project in days rather than weeks, making it invaluable for modern software engineering teams under tight deadlines.

Efficiently integrating new developers into existing teams and projects is one of the most persistent challenges in software development. The Onboarding Manager addresses this by automating environment configuration, centralizing documentation, and aiding codebase comprehension. This reduces the dependency on senior team members and accelerates the time-to-productivity. Ultimately, the Onboarding Manager enhances team efficiency, developer satisfaction, and long-term retention by providing a scalable, standardized solution to the onboarding problem.

References

[1] J. Phelan, "Onboarding New Employees — Without Overwhelming Them," Harvard Business Review, Apr. 02, 2024. https://hbr.org/2024/04/onboarding-new-employees-without-overwhelming-them

[2] "Developer Retention: The Costs of Onboarding Software Developers - Growin," May 13, 2022. https://www.growin.com/developer-retention-costs-onboarding

[3] "Developer onboarding: Tools to make the process fast and fun | garden.io," Garden.io, 2023. https://garden.io/blog/developer-onboarding (accessed Dec. 17, 2024).

[4] Whatfix, 2024. https://go.whatfix.com/lp/why-whatfix/brand/3/?utm_source=google&utm_medium=cpc&utm_campaign=Brand-US&utm_content=brand-phrase&utm_term=%7Bkeyword%7D&gad_source=1&gclid=Cj0KCQiAvP-6BhDyARIsAJ3uv7beGhlJOwlyGMq-hord4AzleCkbJMXNgBiGSbXbq-jYxwt44u3XOVUaAl0zEALw_wcB (accessed Dec. 17, 2024).