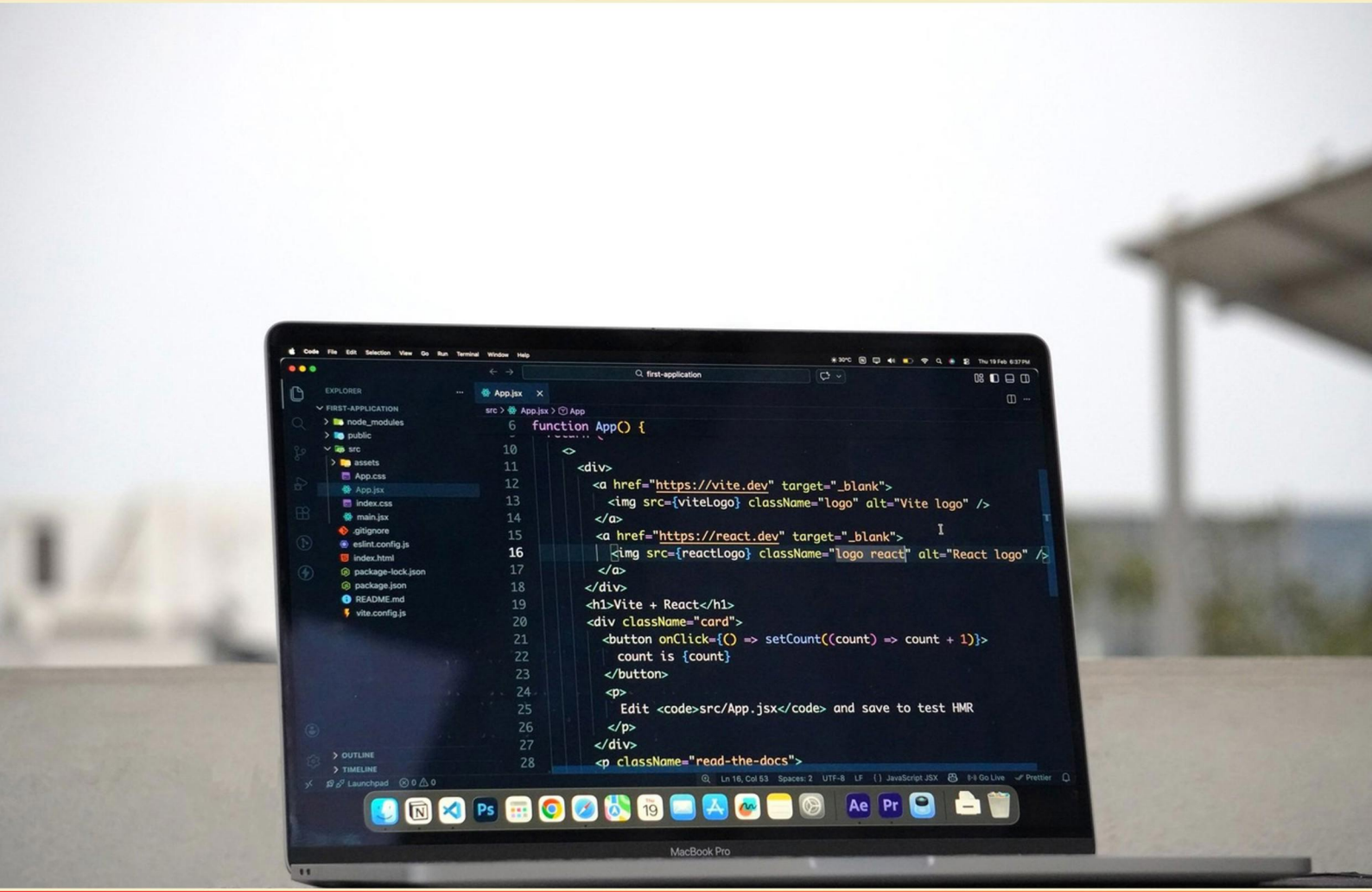
GITLAB CERTIFIED ASSOCIATE PRACTICE EXAM (SAMPLE)

STUDY GUIDE



SAMPLE STUDY GUIDE WITH LIMITED QUESTIONS

VISIT US ONLINE FOR THE FULL VERSION STUDY GUIDE

https://gitlabcertifiedassoc.examzify.com

Table of Contents

Introduction

Preparing for a certification exam can feel overwhelming, but with the right tools, it becomes an opportunity to build confidence, sharpen your skills, and move one step closer to your goals. At Examzify, we believe that effective exam preparation isn't just about memorization, it's about understanding the material, identifying knowledge gaps, and building the test-taking strategies that lead to success.

This guide was designed to help you do exactly that.

Whether you're preparing for a licensing exam, professional certification, or entry-level qualification, this book offers structured practice to reinforce key concepts. You'll find a wide range of multiple-choice questions, each followed by clear explanations to help you understand not just the right answer, but why it's correct.

The content in this guide is based on real-world exam objectives and aligned with the types of questions and topics commonly found on official tests. It's ideal for learners who want to:

- Practice answering questions under realistic conditions,
- Improve accuracy and speed,
- Review explanations to strengthen weak areas, and
- Approach the exam with greater confidence.

We recommend using this book not as a stand-alone study tool, but alongside other resources like flashcards, textbooks, or hands-on training. For best results, we recommend working through each question, reflecting on the explanation provided, and revisiting the topics that challenge you most.

Remember: successful test preparation isn't about getting every question right the first time, it's about learning from your mistakes and improving over time. Stay focused, trust the process, and know that every page you turn brings you closer to success.

Let's begin.

How to Use This Guide

This guide is designed to help you study more effectively and approach your exam with confidence. Whether you're reviewing for the first time or doing a final refresh, here's how to get the most out of your Examzify study guide:

1. Start with a Diagnostic Review

Skim through the questions to get a sense of what you know and what you need to focus on. Your goal is to identify knowledge gaps early.

2. Study in Short, Focused Sessions

Break your study time into manageable blocks (e.g. 30 – 45 minutes). Review a handful of questions, reflect on the explanations.

3. Learn from the Explanations

After answering a question, always read the explanation, even if you got it right. It reinforces key points, corrects misunderstandings, and teaches subtle distinctions between similar answers.

4. Track Your Progress

Use bookmarks or notes (if reading digitally) to mark difficult questions. Revisit these regularly and track improvements over time.

5. Simulate the Real Exam

Once you're comfortable, try taking a full set of questions without pausing. Set a timer and simulate test-day conditions to build confidence and time management skills.

6. Repeat and Review

Don't just study once, repetition builds retention. Re-attempt questions after a few days and revisit explanations to reinforce learning. Pair this guide with other Examzify tools like flashcards, and digital practice tests to strengthen your preparation across formats.

There's no single right way to study, but consistent, thoughtful effort always wins. Use this guide flexibly, adapt the tips above to fit your pace and learning style. You've got this!

Questions

1. What is one way to commit changes in GitLab?

A. Only through the command line

B. By merging directly into the main branch

C. Using the GUI with one commit per file

D. By creating a new project every time

2. How frequently should CI/CD pipelines be run according to best practices?

A. Once a week

B. On every push, merge request, or scheduled intervals

C. Only when a project is completed

D. At the end of each month

3. What is the main purpose of Continuous Integration (CI) in GitLab?

A. To automate deployment processes

B. To integrate code changes in a shared repository

C. To conduct security audits of the code

D. To increase the number of releases

4. What is the default stage for scanning jobs in GitLab?

A. Build

B. Test

C. Deploy

D. Cleanup

5. What is a local repository in Git?

A. A remote copy of the repository

B. A local version of the upstream repository

C. A backup of all branches

D. A cloud-based source for project data

6. What should be set for the allow_failure parameter in scanning jobs?

A. Set to false to ensure strict checks

B. Set to true to prevent pipeline blockage

C. Set to default for simplicity

D. Set to conditional based on job requirements

7. Which feature of GitLab Review Apps facilitates collaboration?

A. Automatic Live Preview

B. Cluster management

C. Package auditing

D. Container orchestration

8. What is a core advantage of GitLab's open-source model?

A. Limited community involvement

B. Free access to all features

C. Inviting contributions from a wider developer community

D. A more expensive licensing model

9. What is the primary function of GitLab Review Apps?

A. To create static web pages

B. To automatically generate dynamic environments for merge requests

C. To manage user permissions in repositories

D. To document project planning and iteration

10. Which of the following statements best describes the forking process in GitLab?

A. It allows you to copy the repository without changes

B. It creates a personal copy to make independent changes

C. It merges multiple repositories into one

D. It restricts collaboration to the original repository

Answers

1. C
2. B
3. B
4. B
5. B
6. B
7. A
8. C
9. B
10. B

Explanations

1. What is one way to commit changes in GitLab?

A. Only through the command line

B. By merging directly into the main branch

C. Using the GUI with one commit per file

D. By creating a new project every time

Committing changes in GitLab can indeed be accomplished using the graphical user interface (GUI), which allows users to make modifications visually. This approach is often more accessible for those who may not be comfortable using command line commands. The GUI provides options for making individual commits for each file, which helps in organizing changes purposefully, making it clearer for tracking purposes and history management in the repository. This method is particularly useful in collaborative environments where distinction between changes made by different contributors can be significant. Using the GUI provides a user-friendly context to preview the changes and selectively commit those that are ready, enhancing workflow efficiency. This is especially advantageous in situations where specific changes need to be committed without affecting other modifications that are still being worked on. Other approaches, such as those where only the command line is utilized or creating a new project for each set of changes, could limit ease of use or introduce unnecessary complexity. For instance, merging directly into the main branch without proper code reviews or testing can also lead to unstable code, which is typically discouraged in practice.

2. How frequently should CI/CD pipelines be run according to best practices?

A. Once a week

B. On every push, merge request, or scheduled intervals

C. Only when a project is completed

D. At the end of each month

Running CI/CD pipelines on every push, merge request, or at scheduled intervals aligns with best practices in DevOps and modern software development. This approach ensures that code changes are continuously integrated and validated, allowing for faster feedback if any issues arise. By triggering the pipelines on every push or merge request, teams can detect and fix problems early in the development cycle, which promotes higher code quality and reduces integration issues. Additionally, scheduling regular intervals for pipeline execution can ensure that any ongoing changes are integrated and tested, even if they aren't tied to a specific commit or request. This frequency supports an agile workflow, helps maintain the stability of the codebase, and encourages collaboration among team members by ensuring that the latest changes are always merged and tested. Adhering to this practice also reduces the risks associated with large deployments that typically occur when code is only integrated at the end of a development cycle, either at project completion or month-end. Frequent testing and integration facilitate a more seamless deployment process and higher confidence in software releases.

3. What is the main purpose of Continuous Integration (CI) in GitLab?

A. To automate deployment processes

B. To integrate code changes in a shared repository

C. To conduct security audits of the code

D. To increase the number of releases

The main purpose of Continuous Integration (CI) in GitLab is to integrate code changes into a shared repository frequently. This process allows developers to merge their individual changes back to the main branch multiple times a day, ensuring that all code alterations are continuously tested and validated. By doing this, CI helps in identifying and resolving issues early in the development cycle, improving code quality and reducing integration challenges that could arise if changes were made in isolation. Continuous Integration facilitates collaboration among team members by providing a streamlined process for combining everyone's contributions. It typically involves automated testing, which enables developers to receive instant feedback on their changes, ensuring that any defects are captured and addressed promptly. The overarching goal of CI is to make the integration of changes seamless and efficient, allowing teams to focus on building features and fixing issues rather than struggling with complex code merges and conflicts. While automated deployment, security audits, and increasing release counts are important aspects of modern development workflows, they are not the primary focus of Continuous Integration itself. CI serves as a foundational practice that enables these other processes to occur more effectively within the development lifecycle.

4. What is the default stage for scanning jobs in GitLab?

A. Build

B. Test

C. Deploy

D. Cleanup

In GitLab CI/CD, the default stage for scanning jobs is the Test stage. This means that when you create a new CI/CD pipeline, any jobs set to run scans, such as security scans or linting jobs, are automatically assigned to the Test stage unless specified otherwise in the configuration. The Test stage is commonly used to verify the integrity and functionality of the code, which aligns perfectly with the purpose of scanning jobs that assess code quality, security vulnerabilities, and compliance. While other stages like Build, Deploy, and Cleanup serve their own specific purposes, the Test stage is particularly geared towards validating changes before they are either built or deployed. Consequently, jobs meant for inspection or verification naturally fall under this category, emphasizing its role in the software development lifecycle.

5. What is a local repository in Git?

A. A remote copy of the repository

B. A local version of the upstream repository

C. A backup of all branches

D. A cloud-based source for project data

A local repository in Git refers to a version of the repository that resides on a developer's local machine. It contains all the project's files, history, and branches, enabling developers to work on their changes without needing constant access to the internet or a remote server. This local setup allows for quick modifications, commits, and browsing through the repository history, fostering a flexible development process. Having a local repository means that a developer can effectively manage and track changes in their codebase, experiment with new features, or make fixes independently. When ready, changes can then be pushed to the upstream repository, which serves as the central or shared location for collaboration with others. The other options misrepresent the concept of a local repository. A remote copy refers to the repository hosted on a server, not on a developer's local machine. While a local repository does contain branches, it is not accurate to define it solely as a backup; it's an active space for development rather than just a safety mechanism. Lastly, a cloud-based source suggests storage on an online platform, which is distinct from the idea of a local repository that resides on a user's local device.

6. What should be set for the allow_failure parameter in scanning jobs?

A. Set to false to ensure strict checks

B. Set to true to prevent pipeline blockage

C. Set to default for simplicity

D. Set to conditional based on job requirements

Setting the allow_failure parameter to true for scanning jobs is beneficial to prevent pipeline blockage. This configuration allows the pipeline to continue running even if the scanning job fails, which means that other jobs in the pipeline can be executed without waiting for the scanning job to pass. This can be particularly important in scenarios where the scanning jobs might be prone to false positives or where organizations prioritize rapid development and delivery over strict compliance with security checks at every stage. For instance, in a CI/CD environment where quick iterations and deployments are essential, allowing a failure in scanning jobs ensures developers can still move forward with the rest of their work without being hindered by scan results that do not impact the immediate functionality or security posture of the deployment. This method can help teams adopt a more agile approach while balancing the need for security vigilance. In contexts where immediate action isn't necessary, or if further investigation into the scan results is required, having the ability to proceed can streamline development processes while still keeping the scanning jobs available for review after the fact.

7. Which feature of GitLab Review Apps facilitates collaboration?

A. Automatic Live Preview

B. Cluster management

C. Package auditing

D. Container orchestration

The feature that facilitates collaboration among team members in GitLab Review Apps is the Automatic Live Preview. When developers create a merge request, Automatic Live Preview allows stakeholders to view and interact with the application as it would appear in production. This immediate visibility helps teams gather feedback quicker and more effectively, enabling better collaboration during the review process. With Automatic Live Preview, reviewers can see the actual changes made in the code without needing to run the application in their local environments. This helps eliminate discrepancies between what the developer sees and what the reviewer experiences. By providing a shared view of the application at various stages of development, it enhances discussions around specific features or bugs, making the review process more efficient and focused. In contrast, cluster management, package auditing, and container orchestration serve other important functions within DevOps workflows, but they do not directly enhance collaboration in the same way that Automatic Live Preview does. Cluster management relates to handling groups of servers, package auditing focuses on security through managing dependencies, and container orchestration pertains to automated management of containerized applications. While these are vital for software deployment and maintenance, they do not specifically enhance real-time collaboration among team members during the review process.

8. What is a core advantage of GitLab's open-source model?

A. Limited community involvement

B. Free access to all features

C. Inviting contributions from a wider developer community

D. A more expensive licensing model

A core advantage of GitLab's open-source model is that it invites contributions from a wider developer community. This allows for collaborative development, as anyone can contribute improvements, features, or fixes to the software. Contributors can include independent developers, organizations, or even companies that use GitLab. This community-driven approach enhances the quality and innovation of the software, as diverse perspectives and expertise are brought to the table. Additionally, the open-source model fosters transparency and trust, as users can inspect the source code, understand how it works, and verify that it meets their security and functional needs. When developers contribute code, they often do so with a vested interest in the sustainability and success of the platform, leading to a robust ecosystem of plugins and integrations that can benefit all users. The other options do not align with the core advantages of open-source software. Limited community involvement contradicts the fundamental principle of open-source collaboration. Free access to all features can be misleading, as while many features may be available, not all may be accessible for free depending on the specific version or configuration. A more expensive licensing model is also contrary to the principles of open-source software, which typically emphasizes accessibility and collaboration over high costs.

9. What is the primary function of GitLab Review Apps?

A. To create static web pages

B. To automatically generate dynamic environments for merge requests

C. To manage user permissions in repositories

D. To document project planning and iteration

The primary function of GitLab Review Apps is to automatically generate dynamic environments for merge requests. This feature allows developers to test new features or changes in a live environment that closely resembles the production setup, providing immediate feedback during the development process. When a merge request is created, a review app can be deployed, allowing team members and stakeholders to view and interact with the changes made. This significantly enhances collaboration and helps in identifying issues early on, leading to more efficient development and higher-quality code. Review apps simplify the testing workflow by providing a consistent and automated way to validate changes in an isolated environment. They are particularly useful for applications that rely on frontend and backend interactions because these apps can simulate the entire stack and allow for proper testing before merging code into the main branch. This practice reduces the risk of conflicts and regressions in the production environment, making it a vital aspect of continuous integration and deployment practices.

10. Which of the following statements best describes the forking process in GitLab?

A. It allows you to copy the repository without changes

B. It creates a personal copy to make independent changes

C. It merges multiple repositories into one

D. It restricts collaboration to the original repository

The forking process in GitLab is best described as creating a personal copy to make independent changes. When a user forks a repository, they create a duplicate of the original repository that they can modify freely without affecting the source project. This is a fundamental feature of GitLab and promotes collaboration, as it allows developers to work on their own versions of projects, experiment, and implement changes without requiring immediate permissions to modify the original code. Forking is particularly useful in open-source projects where individuals or teams may want to make contributions, test features, or refine their ideas before proposing those changes back to the maintainers of the original repository through a merge request. This creates an environment where innovation is encouraged and is beneficial to the collective development efforts. The other choices either misunderstand the concept of forking or provide inaccurate descriptions: copying the repository without changes does not capture the essence of forking since forking is about creating a separate copy for independent work. Merging multiple repositories into one describes a completely different action in version control that is generally conducted via pull requests or merges. Restricting collaboration to the original repository contradicts the very purpose of forking, which is to enhance collaborative efforts by allowing developers to work independently.

Next Steps

Congratulations on reaching the final section of this guide. You've taken a meaningful step toward passing your certification exam and advancing your career.

As you continue preparing, remember that consistent practice, review, and self-reflection are key to success. Make time to revisit difficult topics, simulate exam conditions, and track your progress along the way.

If you need help, have suggestions, or want to share feedback, we'd love to hear from you. Reach out to our team at hello@examzify.com.

Or visit your dedicated course page for more study tools and resources:

https://gitlabcertifiedassoc.examzify.com

We wish you the very best on your exam journey. You've got this!