**T.Y. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 10

**System Review and Acceptance**

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Approved By:**

**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Introduction |  |
| 2 | Review Types |  |
| 3 | Verification Summary |  |
| 4 | Verification Matrix |  |

# INTRODUCTION

*At the time of the scheduled peer review, ensure proper representation and preparation by the reviewers. Provide clarifications on the work products. Present comments and listen to the comments of the other reviewers. Comments can be presented either by page or by reviewer. Keep the comment discussions short with a focus on detection, not correction. Editorial comments are provided separately and are not discussed at the scheduled review.*

*Participate in categorizing comments. The comments will be categorized and documented as errors, defects, and action items. Refer to the definitions for the categorization rules, which are summarized as follows:*

* *Errors (i.e., problems in the material currently under peer review).*

*Optionally, errors are subcategorized as major (affects functionality and/or performance) and minor (does not affect functional- ity and/or performance).*

* *Defects (i.e., problems in materials previously peer reviewed).*

*Optionally, defects are also subcategorized as major and minor.*

*Note: Defects will further be categorized as delivered or undelivered in the program’s change request system.*

* *Action items (i.e., unresolved comments requiring further investigation)*
* *A comment can remain categorized as a comment if the reviewers and presenters agree that there is no error, defect, or action item required.*

*To complete the peer review you must identify errors, defects, and action items to be resolved and documented. If needed, follow the program’s or project’s defined decision-making processes to elevate and reconcile any issues encountered in resolving peer review errors, defects, or action items with appropriate stakeholders. To ensure completion, per- form the following:*

* *Correct all errors and update the peer review information to indicate that the error is resolved.*
* *Submit change request paperwork for all defects. The status and tracking of the defect corrections are then handled through the change request system. The defects associated with the peer review should indicate this transfer and are categorized as resolved, allowing the peer review to be closed.*
* *Resolve and complete all action items. If any action items cannot be completed within the two-week period, these action items should be moved to the program- or project-level action item tracking system. The action items associated with the peer review should indicate this transfer and are categorized as resolved, allowing the peer review to be closed.*

# REVIEW TYPES

*Design and code reviews promise to improve software quality, ensure compliance with standards, and serve as a valuable teaching tool for developers. As with most practices, there are subtle nuances surrounding how they're performed that can dramatically affect their value. In some organizations, reviews are a valuable aspect of the software lifecycle. In others, they are a necessary evil tainted with political bureaucracy and big egos. Suboptimal reviews conducted late in the lifecycle are often misguided due to few objective guidelines that help guide the review process. When used throughout the development lifecycle, code and design quality metrics are valuable inputs to the review process.*

* 1. *Reviews Increase Agility Continuous Integration.*

*Agile practices are abundant, and for many teams interested in increasing their agility, valuable energy and resources have been devoted to improving these practices. Because of this, many teams have abandoned reviews while emphasizing other aspects of agility. But, reviews are an important tool in the agile toolkit.*

*A driving principle of the Agile Manifesto is continuous attention to technical excellence. Another is embracing and harnessing change as an opportunity to increase customer advantage. For developers, change often begins and ends with modifications to the source code. A poorly designed application with smelly code is a breeding ground for risk that makes change incredibly difficult, and is the greatest technical inhibitor to increased agility. Effective reviews that emphasize design quality and code cleanliness are an important aspect of increased agility. Reviews done right help ensure continuous attention to technical excellence. Unfortunately, not all reviews are done right.*

*1.2 Review Worst Practices*

*Some development teams find reviews a healthy and valuable asset to developers and the project team. Other teams realize little value from their review process. There are numerous causes for painful and ineffective reviews. Some symptoms of ineffective reviews include:*

* *Witch hunt reviews - Many reviews degrade quickly into attack and defend mode. This often occurs because the developer who wrote the code feels attacked and threatened when reviewers make direct and opinionated statements about the code. Nothing could be less productive.*
* *Curly brace reviews - Some reviews emphasize formatting and comments instead of more serious problems. Is placement of curly braces and misspelled comments really that important? Curly brace reviews are feeding ground for the anal retentive, and provide no real value.*
* *Blind reviews - Often times, reviewers walk into the review meeting having never laid eyes on the code they are about to review. Most of the review time is spent trying to figure out what the code does. Spending time in the review meeting attempting to understand the code instead of reviewing it for more serious ailments is a waste of time.*
* *Exclusionary reviews - Many times, the code provided for the review is only a sampling of the code written. For example, unit tests might be excluded from the review. In an unhealthy review environment, providing impartial and incomplete code listings will leave the reviewers wondering how the code actually works.*
* *Tree killer review - If you can't baffle them by providing half of what they need to understand the code, then maybe overwhelming them by providing thousands of lines of code might work. Waiting until codebase is incredibly large to host the first review is entirely ineffective. Not only is it to difficult to provide effective feedback on a large codebase, these reviews are often held late in the lifecycle and do not allow the developer to improve her code based on the feedback received.*
* *Token review - It's not uncommon for management to dictate that reviews be held. Token reviews are typically held for political reasons. Management wants to ensure that all code is reviewed for auditing purposes. Unfortunately, developers realize very little value surrounding these reviews. Any problems found are not fixed unless they are absolutely critical. Since the primary motivation is an audit trail for management, the team has little motivation to improve the code.*
* *World review- The reviews conducted with great number of people in attendance. This can be incredibly intimidating for the developers whose code is being reviewed, and it is not sure what value it provides to invite so many people. A few developers, up to five, should serve all the needs required of the review process. If more people want to provide input, there are better ways.*

*The Design checklist is as follows:*

* *Deficiencies and conflicts in requirements, architecture, or program/project plans will be reported.*
* *Design decisions and the decision rationales will be recorded according to plans and defined processes.*
* *Top-level software components of the software end item will be identified and described.*
* *Static relationships between top-level software components will be defined.*
* *Dynamic relationships between top-level software components will be defined.*
* *The concepts of execution of the software end item and its components will be defined.*
* *External interfaces of the software end item and its components will be identified and described.*
* *Top-level software components will be decomposed into lower-level software units.*
* *Internal interfaces between software units will be identified and described according to the standards identified by the project.*
* *Design traceability data will be documented according to plans, processes, and product standards.*
* *Design definitions will be documented according to plans, defined processes, and standards.*
* *Measurement and estimated data will be collected.*
* *Applicable work products will be submitted for peer reviews in accordance with project plans.*
* *Applicable work products will be submitted for control in accordance with program or project plans.*

# VERIFICATION SUMMARY

*Note: The verification summary is required to be written for all the objectives and processes as they were detailed as User Stories. Replicate the standard template for objectives and process for the goals.*

# VERIFICATION STEPS: GOAL-1

|  |  |
| --- | --- |
| Objective-1 | Objective Name |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | *<type of user>* |
| Verification Steps | 1. |
|  | 2. |
|  | 3. |
|  | 4. |
|  | 5. |
|  | 6. |
|  | 7. |
|  | 8. |
|  | 9. |
|  | 10. |

# VERIFICATION STEPS: GOAL-2

|  |  |
| --- | --- |
| Objective-1 | Objective Name |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a** *<type of user>* |
| Verification Steps | 1. |
|  | 2. |
|  | 3. |
|  | 4. |
|  | 5. |
|  | 6. |
|  | 7. |
|  | 8. |
|  | 9. |
|  | 10. |

# VERIFICATION STEPS: GOAL-3

|  |  |
| --- | --- |
| Objective-1 | Objective Name |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a** *<type of user>* |
| Verification Steps | 1. |
|  | 2. |
|  | 3. |
|  | 4. |
|  | 5. |
|  | 6. |
|  | 7. |
|  | 8. |
|  | 9. |
|  | 10. |

# VERIFICATION STEPS: GOAL-4

|  |  |
| --- | --- |
| Objective-1 | Objective Name |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a** *<type of user>* |
| Verification Steps | 1. |
|  | 2. |
|  | 3. |
|  | 4. |
|  | 5. |
|  | 6. |
|  | 7. |
|  | 8. |
|  | 9. |
|  | 10. |

# VERIFICATION STEPS: GOAL-5

|  |  |
| --- | --- |
| Objective-1 | Objective Name |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a** *<type of user>* |
| Verification Steps | 1. |
|  | 2. |
|  | 3. |
|  | 4. |
|  | 5. |
|  | 6. |
|  | 7. |
|  | 8. |
|  | 9. |
|  | 10. |

# VERIFICATION STEPS: GOAL-6

|  |  |
| --- | --- |
| Objective-1 | Objective Name |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a** *<type of user>* |
| Verification Steps | 1. |
|  | 2. |
|  | 3. |
|  | 4. |
|  | 5. |
|  | 6. |
|  | 7. |
|  | 8. |
|  | 9. |
|  | 10. |

# VERIFICATION MATRIX

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| User Story | Step-1 | Step-2 | Step-3 | Step-4 | Step-5 | Step-6 | Step-7 | Step-8 | Step-9 | Step-10 |
| G1:O1 | √ | X |  |  |  |  |  |  |  |  |
| G1:P1 |  |  |  |  |  |  |  |  |  |  |
| G1:P2 |  |  |  |  |  |  |  |  |  |  |
| G1:O2 |  |  |  |  |  |  |  |  |  |  |
| G1:P1 |  |  |  |  |  |  |  |  |  |  |
| G1:P2 |  |  |  |  |  |  |  |  |  |  |
| G2:O1 |  |  |  |  |  |  |  |  |  |  |
| G2:P1 |  |  |  |  |  |  |  |  |  |  |
| G2:P2 |  |  |  |  |  |  |  |  |  |  |
| G2:O2 |  |  |  |  |  |  |  |  |  |  |
| G2:P1 |  |  |  |  |  |  |  |  |  |  |
| G2:P2 |  |  |  |  |  |  |  |  |  |  |
| G3:O1 |  |  |  |  |  |  |  |  |  |  |
| G3:P1 |  |  |  |  |  |  |  |  |  |  |
| G3:P2 |  |  |  |  |  |  |  |  |  |  |
| G3:O2 |  |  |  |  |  |  |  |  |  |  |
| G3:P1 |  |  |  |  |  |  |  |  |  |  |
| G3:P2 |  |  |  |  |  |  |  |  |  |  |
| G4:O1 |  |  |  |  |  |  |  |  |  |  |
| G4:P1 |  |  |  |  |  |  |  |  |  |  |
| G4:P2 |  |  |  |  |  |  |  |  |  |  |
| G4:O2 |  |  |  |  |  |  |  |  |  |  |
| G4:P1 |  |  |  |  |  |  |  |  |  |  |
| G4:P2 |  |  |  |  |  |  |  |  |  |  |
| G5:O1 |  |  |  |  |  |  |  |  |  |  |
| G5:P1 |  |  |  |  |  |  |  |  |  |  |
| G5:P2 |  |  |  |  |  |  |  |  |  |  |
| G5:O2 |  |  |  |  |  |  |  |  |  |  |
| G5:P1 |  |  |  |  |  |  |  |  |  |  |
| G5:P2 |  |  |  |  |  |  |  |  |  |  |
| G6:O1 |  |  |  |  |  |  |  |  |  |  |
| G6:P1 |  |  |  |  |  |  |  |  |  |  |
| G6:P2 |  |  |  |  |  |  |  |  |  |  |
| G6:O2 |  |  |  |  |  |  |  |  |  |  |
| G6:P1 |  |  |  |  |  |  |  |  |  |  |
| G6:P2 |  |  |  |  |  |  |  |  |  |  |