**ISSUES IDENTIFICATION AND TRACKING DOCUMENT**

**Introduction**

The Issue Identification and Tracking Document is an important tool for any project. This document is also commonly used during the post go-live phase of project/product implementation. Inevitably, all projects encounter issues which must be documented and, ultimately, resolved. More complex projects often result in a higher number of issues which can adversely impact the project if not managed carefully. The Issue Identification and Tracking Document allows the project manager and team to capture issues as they are identified. Additionally, it allows the team to effectively manage these issues by prioritizing them, assigning them to various team members/stakeholders for action, and seeing them through to resolution

**Explanation of Issues Identification and Tracking Document Contents**

Formats and contents for the Issues Identification and Tracking Document may vary based on system requirements or organizational standards. Most Issues Identification and Tracking Documents consist of fundamental contents which may be applied across a wide range of projects or software/system types. This section will provide explanations for each section of the Issues Identification and Tracking Document.  
  
Issue #: Each issue should be sequentially numbered for reference and tracking purposes.

Issue Description: This section should provide a description of each issue to include what the potential impact will be and what part of the project or system will be affected.

Issue Type: This section should identify the type of issue that has been identified. Common issue types are: technical issues, resource issues, process issues, or external issues—those which fall outside of the organization. This categorization aids in assigning an issue owner

Identified By: This section should provide the name(s) of the person/people who identified the issue. This is helpful in the event more clarification is needed as the issue is assigned or moves toward resolution.   
  
Status: Each issue’s status should be updated throughout the resolution process. As this is a living document, each issue’s status should be updated as any changes occur.

Resolution Description: This section should describe what was done to resolve the identified issue. This is important as lessons learned are archived for system deployments, projects, or go-live implementations. Future projects may encounter similar issues in which case the project teams can identify resolutions that were implemented in the past.

|  |  |  |
| --- | --- | --- |
| Issue# | Issue Description | Resolution Description |
| 1. | Trains can collide on the same track | Trains should get information about the other trains from the dispatcher or the other trains themselves |
| 2. | Time lag from the trains can leads to accidents | Before the train starts from the station it should get a detailed log about the trains, their status. |
| 3. | What if tracks do not respond to the vaults on time to change directions | If the dispatcher do not change the direction vault of the track, the time to reach the destination lags. We are working on it if it could be done manually |
| 4. | What if data in the fault log is corrupted and the trains delay or accidents occur due to that | We are working on it to make sure that fault log is automated according to the situation by taking information from the trains |
| 5. | What if the dispatcher who views all the controls and can stop the train or handle the situation is not present for some time at the dispatcher station. | We r working on it and have no idea how to automate the dispatcher station |
| 6. | Data storage about the trains and the particular situation goes on increases day to day .so how to handle the data flow | We can reduce the data flow by only storing the particular necessary data for a long time and normal data for a short period of time |
| 7. | Division of work is a bit difficult in our team | As everyone in the team are new to the work and are not specialized for that work ,division of work is difficult and we are trying to overcome it by choosing work according to our interest. |
| 8. | Use of database | Initially we thought we can implement using sql database inorder to store train and tracks information.later on because of less amount of data we implemented using text file.one separate text file for trains tracks and routes |
| 9. | Implementation using swing | It was difficult to relate swing components with java classes . |

March 13 – march 17:while implementing multiple trains we got many coordination issues.how multiple trains should move with out colliding with each other.

In this week we tried to solve the issue by blocking the entire the route for a particular train to move from source to destination.

March 17 –march 22: while implementing trains we got a problem to know the location of particular trains and its details.to solve it we created a dynamic table which gives details of trains.

March 18:we got issues while integrating code manually.we solved it by integrating code on github.

April 1:problem:implementing the freight train and avoiding collision with passenger train

Solution: implementing the freight train includes more functionalities like maintaining separate route and according to situation moving other passenger trains to implement freight train. We have removed the freight train due to time factor. We do not have enough time to implement functionalities of the freight train.

April 10:problem: we have identified that the passenger trains are colliding with each other

Solution:we have changed the functionalities like if a particular section is blocked for other train ,then the train moving to that section takes a pause or halt and waits till the section is free.