**QA docs 8 Summary**

**Software configuration management**

GitLab will allow a user to:

1. Retrieve copies of any version of a file, enabling recovery of previous or ‘old’ versions.

2. Retrieve copies of any version of a directory structure, with its files, enabling recovery of previous versions of software, or snapshots of documentation.

3. Check in changes to the file, causing the changes to be recorded.

4. Inquire about differences between versions, obtain a log summarising the changes checked in for a particular version and produce a history of a file showing all changes and the users responsible.

Files needed:

1. List of project deliverables
2. Design specification, including any accompanying diagrams.
3. Test specification.
4. User interface document, including any on-screen presentations.
5. Maintenance manual.
6. End-of-project report.
7. Source code and tools (e.g., build files)

**Configuration Items, References and Status**

Configuration items are project items which are controlled by the configuration management system.

The QA Manager must compile a list of project deliverable items that will be kept under configuration management, along with a name and location of the item. It will be in a file called config\_refs in the config directory

Any item included must have one of the following statuses:

1. Draft - the document is currently under development.
2. For review - the document is ready for formal review.
3. Release - the document has successfully passed its review and is thus complete and correct

**Directory Structures**

This contains submitted documents produced by the group members. Each document should be in its own directory. They should be submitted once they are ready to be reviewed.

* config. Management documents are stored in this directory. Specifically, there should be a minutes directory (see Section 2.4.3, “Managing Minutes of Meetings” below), a blogs directory (see Section 2.4.4, “Managing blogs” below) and a configuration file.
* src. This contains the source code for the project, including any module tests, arranged into the relevant directories corresponding to their position in the package hierarchy or in other language defined arrangements. If a project requires more than one program in the system, one sub-directory, appropriately named, may be required for each program. Source code should be under configuration control once it is part of the designed system. Before that, it should be put under the dev folder.
* dev. This directory should contain a folder for the date of each tutorial. If there is a tutorial on 14th February 2021, a folder named 20210214 should be created. This format ensures that folders are listed correctly by date in the dev folder. Where a student creates a draft document or some code as part of their duties that week, it should be submitted to the folder for the next tutorial, so that all group members can see it

**Managing Documents**

Team members will all have working copies of the repository. They are responsible for ensuring that they have the correct version and do not over-write changes that took place after they took a copy of the repository. This will often involve informing other group members that they are working on a specific document or piece of code. Major changes may take place through version control branch.

**PROBLEM REPORTING AND CORRECTIVE ACTION**

The project can use the issue tracking system provided with GitLab for assigning weekly tasks to group members. When the group member commits files related to the issue, they should refer to the issue number in their commit message, so that the group can see what has been done in response to the issue

Problems relating to Released items in the docs directory should go through a slightly more formal problem reporting and action process. When an issue is raised with such a document, then the person assigned to make the change to the document should be recorded and the issue moved to the Doing board. When they have made the change to the document, they should change its status to For Review, and move it to the Reviewing board. The QA Manager should check that the changes are appropriate (organising a formal review if they judge it necessary) and check out the document and change its status to Release, and mark the issue as “Done”. The Issue Number of the Issue raised should be listed in the document history of the document as the reason for making a new version.