**Software Project Documentation**

Contents

[1. Introduction 3](#_Toc512345213)

[1.1. Objectives and scope 3](#_Toc512345214)

[1.2. Terms, abbreviation and definitions 3](#_Toc512345215)

[2. General concepts 3](#_Toc512345216)

[2.1. Documentation goals 3](#_Toc512345217)

[2.2. Documentation development process 3](#_Toc512345218)

[3. Software project documents 3](#_Toc512345219)

[3.1. Roadmap 3](#_Toc512345220)

[3.2. Development 4](#_Toc512345221)

[3.2.1. Requirements 4](#_Toc512345222)

[3.2.2. Process 4](#_Toc512345223)

[3.2.3. Getting started 4](#_Toc512345224)

[3.2.4. Validation, Verification and Testing 5](#_Toc512345225)

[3.2.5. Coding Conventions 5](#_Toc512345226)

[3.2.6. Technical documentation 5](#_Toc512345227)

[3.3. Maintenance & Support 6](#_Toc512345228)

[3.3.1. Infrastructure 6](#_Toc512345229)

[3.3.2. Deployment guide 6](#_Toc512345230)

[3.3.3 Continuous integration Guide 6](#_Toc512345231)

[3.4. Reports and metrics 6](#_Toc512345232)

[3.5. User documentation 6](#_Toc512345233)

# Introduction

## Objectives and scope

This guideline is aimed to define standard recommendations and best practices, which have to be followed when performing documentation on the project.

Target audience is all team members.

## Terms, abbreviation and definitions

|  |  |
| --- | --- |
| **Term** | **Description** |
| ER | Entity Relationship |
|  |  |
|  |  |

# General concepts

## Documentation goals

The main goal of effective documentation is to ensure that developers and stakeholders are headed in the same direction to accomplish the objectives of the project. It helps to track activities related to the project, find out whether time constraints are being met, monitor productivity and plan. Document helps every team member that is involved in project development to have complete knowledge of their responsibilities, have a clear idea of what is expected from them and how they need to manage their work.

## Documentation development process

High-Level Requirements and Design Documents

Detailed Requirements and Design Documents

Time

1. **Planning**

(stakeholders, the software team; architects, UX designers, developers)

**What**

**How**

**ER Diagram**

**UML Diagrams**

**UX Layouts**

Test Plans

Test Documentation

**2. Testing**

(QA Engineers)

**How to Test**

**What to Test**

Project/iteration Start

Project/Iteration Finish

Maintenance & Support guides

User Manuals

Reports & Metrics

**3. Maintenance**

**(Devops, Admins, End-users)**

**4. Handover**

**How to use it**

**How to maintain, restart, install it**

# Software project documents

## Roadmap

A project roadmap is an extremely detailed picture of a project’s deliverables. It gives a clear overview of what is expected to happen at certain times during the project life cycle. A project management roadmap is an important tool as it allows stakeholders to keep track of project milestones and helps with the coordination and communication of different elements of the project. Typical project roadmap indicates project timelines. It also clearly itemizes the risks involved in the project, what impact the project would make, highlights the project specifications, communicates where the team is headed – helping to align their daily tasks with the project’s overall goals and defines success factors. However, it should be noted that the project plan is subject to change during the course of the project.

## Development

### Requirements

A requirements document provides information about the system functionality, what a system should do. It contains UX designs, business rules, user stories, use cases, etc.

The best practice is to write a requirement document using a single, consistent template that all team members adhere to.

Here are the main recommendations to follow:

* Roles and responsibilities. Start your document with the information about project participants including a product owner, team members, and stakeholders. These details will clarify responsibilities and communicate the target release goals for each of the team members.
* Team goals and a business objective. Define the most important goals in a short point form.
* Background and strategic fit. Provide a brief explanation about the strategic aim of your actions. Why are you building the product? How do your actions affect the product development and align with company’s goals?
* Assumptions. Create a list of technical or business assumptions that the team might have.
* User Stories. List or link user stories that are required for the project. A user story is a document written from the point of view of a person using your software product. The user story is a short description of customer actions and results they want to achieve.
* User interaction and design. Link the design explorations and wireframes to the page.
* Questions. As the team solves the problems along the project progression, they inevitably have many questions arising. A good practice is to record all these questions and track them.
* Not doing. List the things which you aren’t doing now but plan on doing soon. Such a list will help you organize your teamwork and prioritize features.

### Process

A process document provides information about the process which team should follow during feature, bugs processing. This document can be a confluence page with described steps and/or diagram with visual representation of this steps.

### Getting started

This document provides information about the necessary environment tools to run application on team member computer and begin work.

### Validation, Verification and Testing

This document contains different types of testing documents (Test strategy, Test plan, Test case specifications, Test checklists).

A **test strategy** is a document that describes the software testing approach to achieve testing objectives. This document includes information about team structure and resource needs along with what should be prioritized during testing. A test strategy is usually static as the strategy is defined for the entire development scope.

A **test plan** usually consists of one or two pages and describes what should be tested at a given moment. This document should contain the list of features to be tested, testing methods, timeframes, roles and responsibilities (e.g. unit tests may be performed either by the QA team or by engineers).

A **test case** specifications document is a set of detailed actions to verify each feature or functionality of a product. Usually, a QA team writes a separate specifications document for each product unit. Test case specifications are based on the approach outlined in the test plan. A good practice is to simplify specifications description and avoid test case repetitions.

**Test checklist** is a list of tests that should be run at a particular time. It represents what tests are completed and how many have failed. All points in the test checklists should be defined correctly. Try to group test points in the checklists. This approach will help you keep track of them during your work and not lose any.

### Coding Conventions

This document attempts to explain the styles and patterns used in the project code. Norms described in document should be followed for new code or existing code.

Here are the main recommendations to follow:

* Code changes. Describe how we work with legacy code, how we work with new code, what should be done if we need to change configuration files etc.
* Clean code principles. Describe basic good code principle what is possible apply to all situations for example KISS, YAGNI etc.
* Internal code principles. Describe rules what team agreed to use in code.
* UnitTests code principles. Describe rules what team agreed to follow during tests development.

### Technical documentation

Software architecture design documents include the main architectural decisions. This document focus on the most relevant and challenging ones. An effective design and architecture document comprises the following information sections:

* Design document template. Discuss and form a consensus with stakeholders regarding what needs to be covered in the architecture design document before it has been created and use a defined template to map architectural solutions.
* Architecture & Design Principles. Underline the guiding architecture and design principles with which you will engineer the product. For instance, if you plan to structure your solution using microservices architecture, don’t forget to specifically mention this.
* Solution details. Describe the contemplated solution by listing planned services, modules, components, and their importance.
* Diagrammatic representation of the solution. Identify the diagrams that need to be created to help understanding and communication of the structure and design principles.

## Maintenance & Support

### Infrastructure

This document attempts to describe a list of all existing environments, describe detailed information about hardware of using servers, explains how they communicate with each other. Contains urls, user accounts etc. Groups all documentation relevant for operational (technical) support of the applications. Ideally this would be enough so a team member without knowledge of the inner workings can offer basic support (like restarting applications, check log files, etc).

### Deployment guide

The deployment guide provides information on the initial installation and configuration of project. This document describes a typical deployment on one or several servers. Introductory information that includes hardware and software prerequisites, architecture information, upgrading from previous installations, and general information about the sample installation.

Instructions on how to install and configure broker hosts and all necessary components and services. Instructions on how to install and configure node hosts and all necessary components and services. Information on how to test and validate an project installation.

### 3.3.3 Continuous integration Guide

The continuous integration guide provides information about process what team decided to use for delivering and sharing they code. What is a branching model they use, how versioning of the code apply etc. Describe in what environments different code versions are to be deployed to (e2e, dev, prod etc.) It’s a good practice to write flow diagrams and share it with all members.

## Reports and metrics

Reports reflect how time and human resources were used during development. They can be generated on a daily, weekly, or monthly basis. Contains process documents such as velocity charts, sprint burndown charts, and release burndown charts.

## User documentation

User documentation is created for product users. It should explain in the shortest way possible how the software can help to solve their problems. Some parts of user documentation, such as tutorials and onboarding, in many large customer-based products are replaced with onboarding training. Nevertheless, there are still complex systems remaining that require documented user guides.

The online form of user documentation requires technical writers to be more imaginative. Online end-user documentation should include the following sections FAQs, Video tutorials, Embedded assistance, Support Portals.

In order to provide the best service for end-users, you should collect your customer feedback continuously. The wiki system is one of the more useful practices. It helps to maintain the existing documentation. If you use the wiki system you won’t need to export documents to presentable formats and upload them the servers. You can create your wiki pages using a wiki markup language and HTML code.