SOFTWARE DEVELOPMENT ROADMAP IN DETAIL

CATEGORIES

Business (https://upplabs.com/category/business/) Clients (https://upplabs.com/category/clients/) Events (https://upplabs.com/category/events/)
FinTech (https://upplabs.com/category/fintech/) Healthcare (https://upplabs.com/category/healthcare/)
Real Estate (https://upplabs.com/category/realestate/) Social contributions (https://upplabs.com/category/social-contributions/)
Technology (https://upplabs.com/category/technology/) **UppLabs (https://upplabs.com/category/upplabs/)**

TABLE OF CONTENTS

Posted by: **UppLabs Team**

Sometimes it's easy to get lost in the abundance of information of the business environment. To visualize the strategic plan of the company, you need a detailed roadmap that can answer the questions "Who?", "What?" and "When?". That's how you can get a clear idea of the processes of upcoming work, the connections between different departments, and outline the project of your potential product.

In this roadmap guide, we collected the detailed information that will help you understand the work of IT infrastructure, estimate the cost and duration of each process and see whether it can support your strategic business needs.

SOFTWARE DEVELOPMENT PROCESS

TAGS

Agile (https://upplabs.com/t
digital metrics (https://upplabs.com/tag/dig
digital security (https://upplabs.com/tag/dig
functional requirements (https://upplabs.com/tag/fur requirements/)
QA (https://upplabs.com/tag
quality assurance (https://upplabs.com/tag/qu assurance/)
quality requirem (https://upplabs.c requirements/)

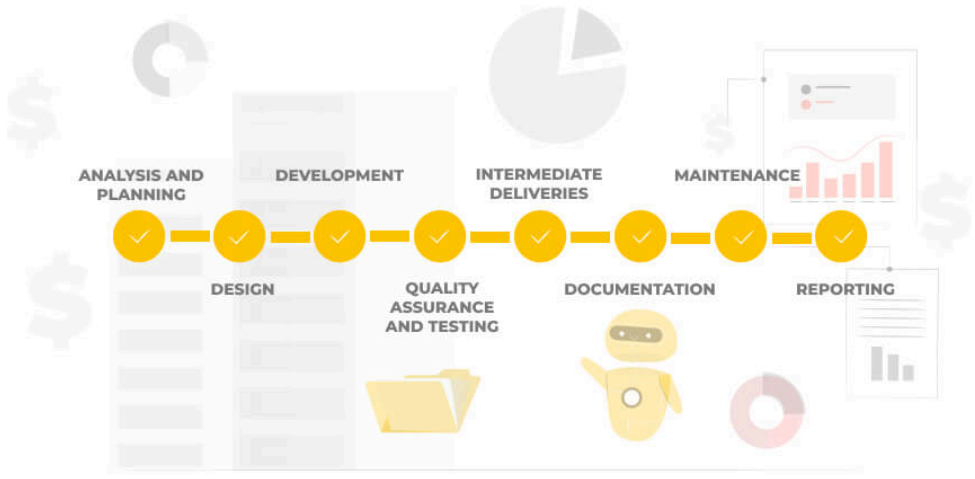
There is a certain difference between the software development roadmap and the IT process roadmap. While the IT process usually is written for product managers and focuses on features, the software development roadmaps are usually created for operation teams and explain the technologies and development operations.

The documentation types, (https://upplabs.com/blog/how-to-prepare-the-documentation-for-successful-software-project-development/)that the team and the client develop their scope, depending on the software development approach. Each is unique in terms of accompanying documentation.

The software development process includes such main stages:

- Analysis and Planning
  – collection of requirements (https://upplabs.com/services/discovery-phase/),
  – research,
  – changes management,
  – risk management (https://upplabs.com/blog/key-elements-of-risk-management/),
  – software architecture.
- Design
  – creating wireframes (https://upplabs.com/services/wireframes-creation/),
  – prototyping,
  – mockups.
- Development
  – backend development,
  – frontend development.
- Quality Assurance and testing
- Intermediate deliveries
- Documentation
- Maintenance
- Reporting

In this guide, we will pay attention to some of the most crucial stages and documents necessary for successful product development.



Software development process

SOFTWARE DEVELOPMENT LIFE CYCLE

The software development life cycle, or SDLC, defines the implementation and support of the product. It helps you turn creative ideas and market demands into functionality and product features.

Depending on your market, project context, and business requirements, you can choose a well-established software development lifecycle model (https://upplabs.com/blog/how-to-build-a-software-product-everybody-would-like-to-use/) or create your own.

MOST POPULAR SDLC MODELS

requirements (https://upplabs.com/tag/rec

requirements specifications (https://upplabs.com/tag/rec specifications/)

roadmap (https://upplabs.com/tag/roa

SDLC (https://upplabs.com/t

software architecture (https://upplabs.com/tag/sof architecture/)

Software Development (https://upplabs.com/tag/sof development/)

software development meth (https://upplabs.com/tag/sof development-methodology

software development proc (https://upplabs.com/tag/sof development-process/)

Software Requirements Spe (https://upplabs.com/tag/sof requirements-specification/

technology (https://upplabs.com/tag/tec

Waterfall (https://upplabs.com/tag/wa

RELATED ARTICLES

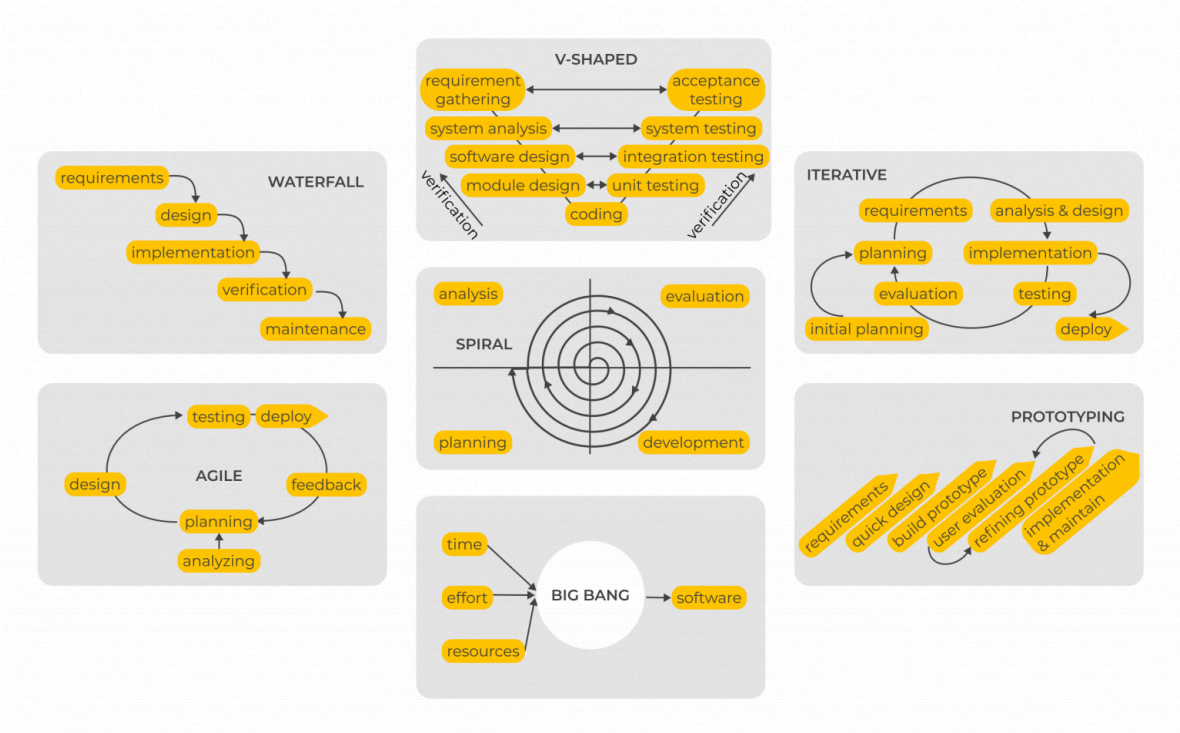
How could digital wallet banking and financial se

(https://upplabs.com/blo digital-wallets-impact-ba financial-services/)

Blockchain and the Meta

(https://upplabs.com/blo and-the-metaverse/)

(https://www.



Types of SDLC

The most popular SDLC Models (sometimes they are called Software Development Methodologies) are:

- Waterfall Model
- V-Shaped Model
- Iterative Model
- Spiral Model
- Big Bang Model
- Agile Model
- Prototyping Model

The waterfall

The waterfall is the first SDLC model in software development history. In the Waterfall model, the development process is linear. Tasks and steps are performed in turns and strict order. Progress flows steadily downward like water over a waterfall.

> It is usually applied to big projects that can be divided into successive logical parts. Besides, no stage can be performed earlier than the previous one. This approach is used in industries such as manufacturing and construction with the linear way of product development.

V-shaped model

The **V-Shaped Model**, also known as the validated model, extends the Waterfall SDLC model. With the V-model, progress doesn't move in a straight line but rises after implementation and coding.

> The project phases in the V-model are the same as in Waterfall, but with validation for each phase of testing. Thus, the V-model is good for the same projects as Waterfall. Its feature is smooth defect tracking as errors are usually detected in the early stages.

Iterative model

The **Iterative Model** combines iterative design and workflow with an incremental build model. In this case, the team develops the product cyclically, evolutionarily creating small parts. A valuable feature of the iterative model is that you can start the development without knowing all the requirements.

> It's good for big and critical projects such as ERP systems, projects with strict requirements for the final product, projects with defined basic requirements, but the ability to develop or improve some features.

Spiral Model

The **Spiral Model** is a combination of Prototyping and Waterfall approaches. The Spiral model includes the same steps as Waterfall, in the same order (requirements gathering, design, implementation, and testing), separated by planning, risk assessment, prototyping, and modeling at each step.

> It's good for complex projects with many small built-in functions, projects with a tight budget, high-risk projects, long-term development projects, projects with no clear requirements on early stages, or projects with a high chance of changes in the development process.

Big Bang SDLC model

The key idea of the **Big Bang SDLC model** is to devote all available resources to the development of the product, mainly in terms of coding, without worrying about fulfilling plans. The development starts with the currently available resources, with very little or no planning. As a result, the customer receives a product that may not even meet the requirements.

> It's good for small teams or individual developers, academic projects, projects with no specific requirements or expected release date, low-risk, repetitive and small projects.

Agile Model

The **Agile Model** is a combination of iterative and incremental approaches focused on adapting to flexible requirements. The main principle of work is the project's division into short cycles (iterations), so it could be easier to receive a particular product at the end of each cycle.

> Agile is effective for implementing big projects. Also, this management methodology is used if the client is constantly changing their expectations. This methodology's advantages are a high level of interaction between project team members, fast results, and flexibility. It's good for almost any type of project, but with a lot of client involvement.

Prototyping Model

The **Prototyping Model** aims to create a working prototype of a software product with limited functionality that can be quickly turned into a finished product. The prototype may not contain the exact logic of the finished product.

> It's good for using in parallel with any other SDLC model, products with a lot of user interactions, and products with early releases. In the prototyping model, user feedback plays a critical role in planning future development.

Download the SDLC comparison table (https://share.hsforms.com/1Xhdur72eRKqJA40Y8SYVpw45lzw)

Choosing the right model greatly defines the expected business product.

TYPES OF DOCUMENTATION

Documents describing the software development process define the requirements of the software, the software design, the development process, and the guarantee of quality. Software development documentation also includes a detailed technical description of the software (software logic, software architecture, data, and storage, etc.).

Document development has five goals:

1. They are a **means of communication** between everyone involved in the development process. Also, they describe the details of decisions made regarding software, design, programming, and testing requirement
2. They **describe the responsibilities** of the development team. They define the roles, taking into account the software, subject, documentation, quality assurance, and everyone involved in the development process;
3. Documentation acts as **checkpoints** that allow managers to evaluate the progress of development. If development documents are missing, incomplete, or out of date, it will be hard to track and control the software project;
4. They **form the basis of the software maintenance documentation** as part of the product documentation;
5. They describe the **history of software development**.

Typical development documents include:

- Development analysis and initial proposals;
- Requirements specifications;
- Functionality specifications;
- User Experience design specifications, including program and data specifications;
- Development plans;
- Plans for the development and testing software;
- Quality assurance plans, standards, and schedules;
- Software architecture design documentation;
- Source code documentation;
- Safety regulations and test information.

Download software development cheat sheet (https://share.hsforms.com/13flg2310RM6v5jLBI8d5OA45lzw)

PRODUCT REQUIREMENT DOCUMENTATION

A Product Requirement Documentation or PRD is a description that includes all the requirements for a specific product and reflects the final outlook of the product. The requirements can be presented both in the form of a single document and a set of documents (layouts, diagrams, documents).

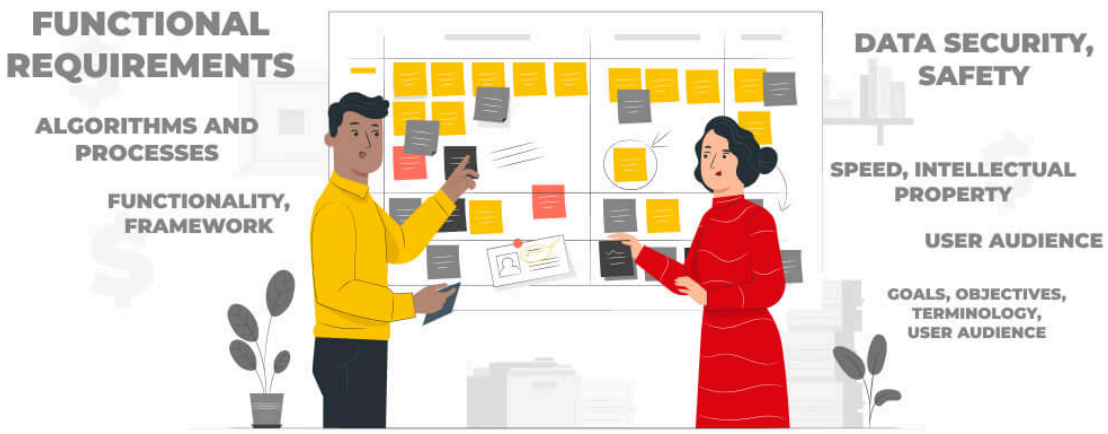
The requirements can be functional and non-functional and usually include:

- a purpose;
- general description;
- specific requirements.

While working on the requirements specification, it is necessary to take into account a large amount of data:

- general block of information about the project: goals, objectives, terminology, user audience;

- general description of the product: functionality, details of users, product operating environment, framework, restrictions, rules, and standards;
- descriptions of algorithms and processes, flow diagrams, functional requirements, interface requirements (UX, API, hardware);
- non-functional requirements ensure data security, safety, speed, intellectual property, and licensing policies.



Requirements specification data

The list of sections may vary depending on the specific type of document (Request for proposal (RFP), Terms of Reference (TOR), Software Requirements Specification (SRS)).

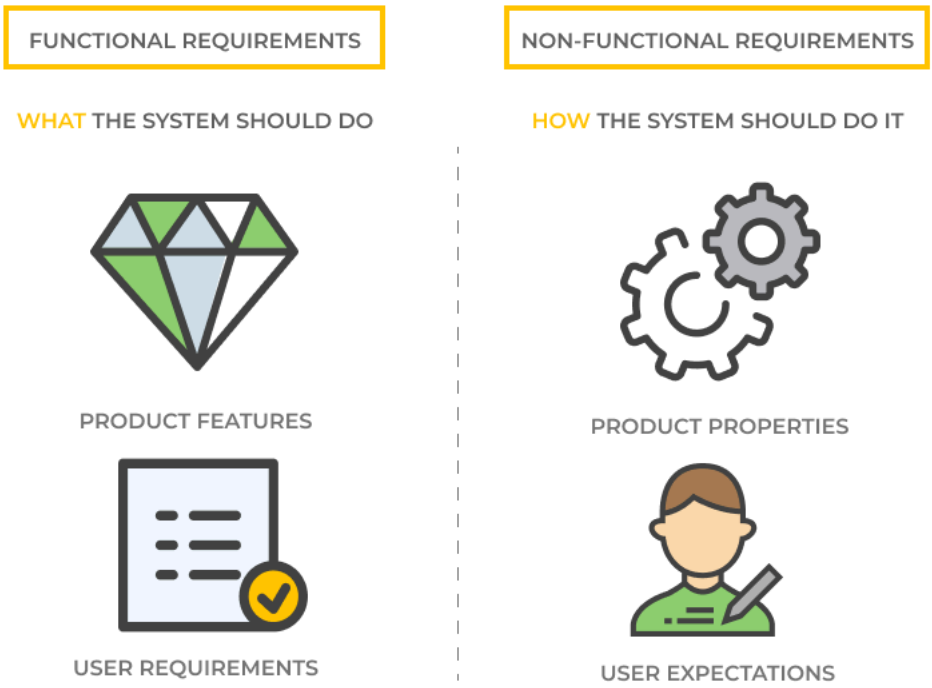
The content and scope of the requirements directly depend on the complexity of the project. For example:

- For **mobile applications** it can be prototyped (possibly interactive), since such applications actively interact with users, and developers need to initially imagine and understand all the processes;
- For **a chatbot,** a functional description is sufficient;
- For **ERP systems**, it should include the most complete information about the future product, which may include prototypes, user stories, data diagrams, etc.

TWO CATEGORIES OF REQUIREMENTS

There are numerous requirements for the characteristics, quality of software products (https://upplabs.com/blog/how-to-control-the-quality-of-your-software-product/), and information systems. Generally, we can divide them into two major categories – functional and non-functional requirements. It is essential to understand the difference between them (https://upplabs.com/blog/the-importance-of-functional-and-non-functional-requirements-in-software-development/).

1. **Functional requirements** describe what specifically needs to be implemented in a system or product, what actions should be performed by users about this development.
2. **Non-functional requirements** describe exactly how the created system or software product works, what properties and characteristics a particular development has.



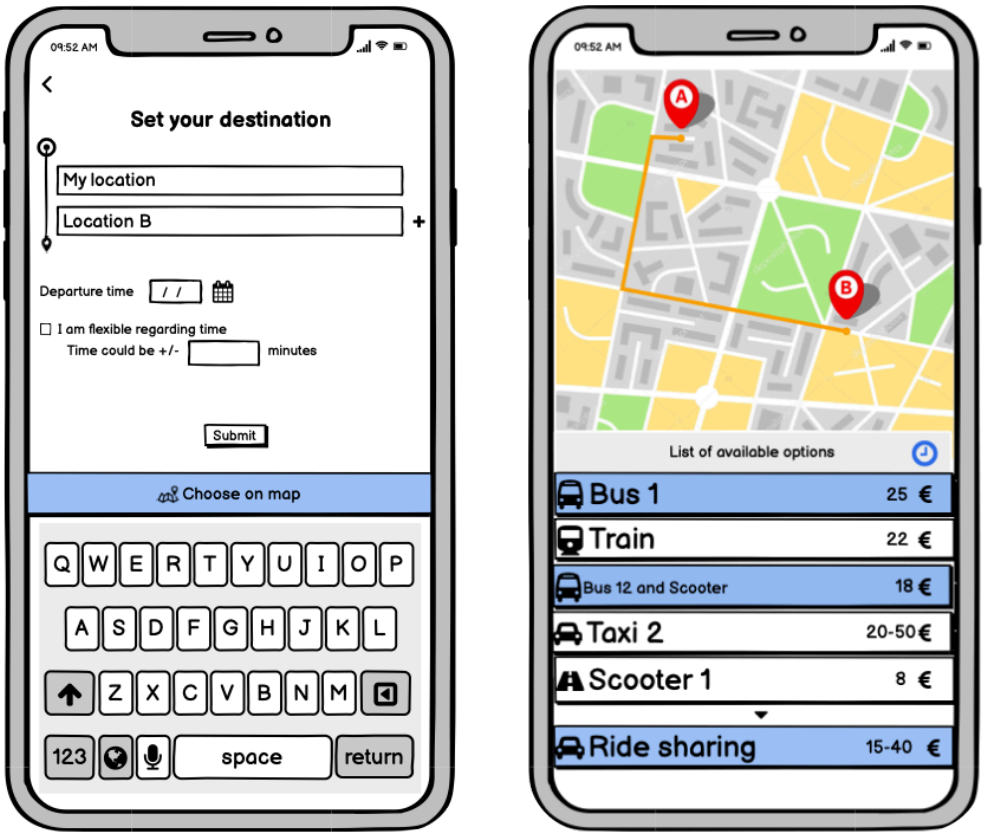
Functional and non-functional requirements. By UppLabs

USER EXPERIENCE DESIGN DOCUMENTATION

User experience design should start at the requirements stage and move on through all the stages of development. The UX design includes research, analysis, prototyping, usability testing, design, and all these stages need to be documented.

Before the developers are ready to start writing their code lines, every project requires a pre-development design phase (https://upplabs.com/services/discovery-phase/) that can show the logic of the website or an app. A UX designer usually uses the deliverables and creates the documentation that the team members can update on the prototyping and designing stage. That's when it's time to make such list documents:

- Wireframes (https://upplabs.com/services/wireframes-creation/) are designed to detail the information, content, and control elements on each interface page. The team uses them in the early development process to set the basic structure before adding the visual design and content.
- Mockups. We can transform the schematic layout into a colorful, bright, and static demonstration of the future product with their help. Mockups allow the clients to see shapes, colors, and fonts.
- A prototype is a high-fidelity interactive performance of the final product. The prototypes often look like actual websites or applications (https://upplabs.com/expertise/web-development/) where users can interact with prototypes and test the features.
- Site maps are HTML pages of a website that contains links to all important pages of the site. A sitemap helps a user quickly find any page on the site, making the minimum number of clicks.
- User flow schemes/user journeys combine web page/interface frameworks (wireframes) and flowcharts to develop User Experience. This method allows you to document the workflow and check the interfaces' design when designing several dynamically changing pages.
- Usability testing reports are used to identify user problems.



Wireframes. By UppLabs

At the final design stages, you should follow all steps as all of them are essential.

SOFTWARE ARCHITECTURE

There are numerous technologies in software development. Some require almost no cost but don't offer high performance; others require significant investments and bring brilliant results. The worst thing you can do is use outdated or unreliable tools that will lead to the code rewrite and extra costs. To avoid this, we recommend that you make some critical decisions.

Check UppLabs case study on the legacy platform rebuild (https://upplabs.com/blog/from-legacy-monolith-app-to-microservices-infrastructure-case-study/)

The software development company needs to work with various databases and APIs and implement scaling and integration with other services. With the help of the latest technology stacks, you can create solutions that streamline your business operations.

A big monolithic application might have 5 or 10 million code lines, which exercises the supporting software in unique ways. Commercial software can be expensive and harder to deploy than open-source.

Monolithic software architecture can be useful if your team is in the early stages, you are building an untested product, and have no experience with microservices. However, microservice architecture is easy to fix and you always know exactly the bugs and the reason for the problem.

In UppLabs (https://upplabs.com/services/web-and-mobile-app-development/), we offer the migration from the monolith to microservices architecture (https://upplabs.com/blog/from-legacy-monolith-app-to-microservices-infrastructure-case-study/). Sometimes it can be considered as a complicated solution, but it is much easier from the point of potential support and scalability.

The Software architecture documentation also includes the goals and objectives of the product, the leading architecture and design principles, user story description, solution details, and diagrams that represent the solution architecture.

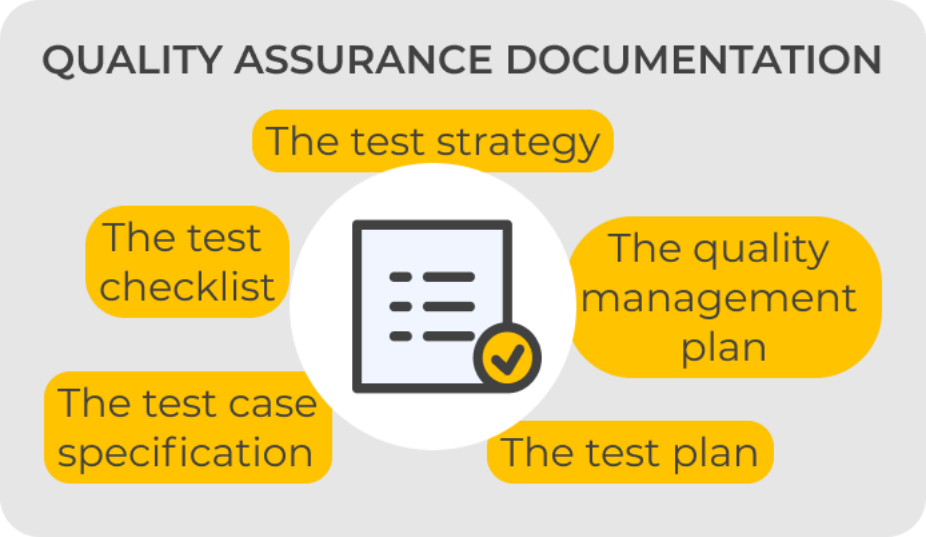
SOFTWARE QUALITY ASSURANCE DOCUMENTATION

Software Quality Assurance (https://upplabs.com/services/quality-engineering-testing/) (SQA) is a set of measures to ensure the quality of software development. The researches (https://www.statista.com/statistics/500641/worldwide-qa-budget-allocation-as-percent-it-spend/) show that 78% of companies use test automation for functional or regression testing while 11% of companies use no test automation. Also, 52% of IT teams sites name the higher QA budgets as the main reason for higher releases of their products.

The SQA works together with the software development lifecycle that regularly verifies the standards of the software at every stage. It prevents potential problems in the early stages.

The quality assurance documentation may include such plans as:

- **The test strategy**
  It's a plan for testing a system or its module, taking into account the specifics of functionality and dependencies of other components.
- **The quality management plan**
  It's a plan and standard to manage product quality in a specific situation that defines the procedures and methods at each level.
- **The test plan**
  It describes the testing procedures that outline the methods, timeframes, roles, and functionality.
- **The test case specification**
  It describes a set of test cases (including their goals, inputs, conditions, steps, and expected results) for a test item (test object).
- **The test checklist**
  It lists several tests that need to be run at different stages.



Quality assurance documentation. By UppLabs

QUALITY MEASURES

The introduction and use of quality metrics (https://upplabs.com/blog/how-to-control-the-quality-of-your-software-product/) are essential for improving control over the development process, particularly over the testing process. If we choose the Bug/Defect Metrics, there are the following types:

- Open/Closed Bugs (the ratio of the number of open bugs to closed (corrected and rechecked)
- Reopened/Closed Bugs (the ratio of the number of reopened bugs to closed (fixed and rechecked)
- Rejected/Opened Bugs (the ratio of the number of rejected bugs to open)
- Bugs by Severity

- Bugs by Priority

Before launching a software product, we need to measure its quality to ensure that it is bug-free. The bugs must be reported per month in all possible environments (Staging, Development, Production). The aim is to reduce the number of bugs on the production every month reviews.

USER DOCUMENTATION

The user documentation should contain the information necessary to use the product. It must fully describe all the functions set in the product description and all user-callable functions of the program.

If the user needs to install the product, then the product installation manual should contain all the necessary information about the product installation. The manual may indicate the minimum and maximum sizes of files to be installed once.

If the user should maintain the product later, then the user documentation should include a program maintenance manual containing all the information regarding the type of maintenance.

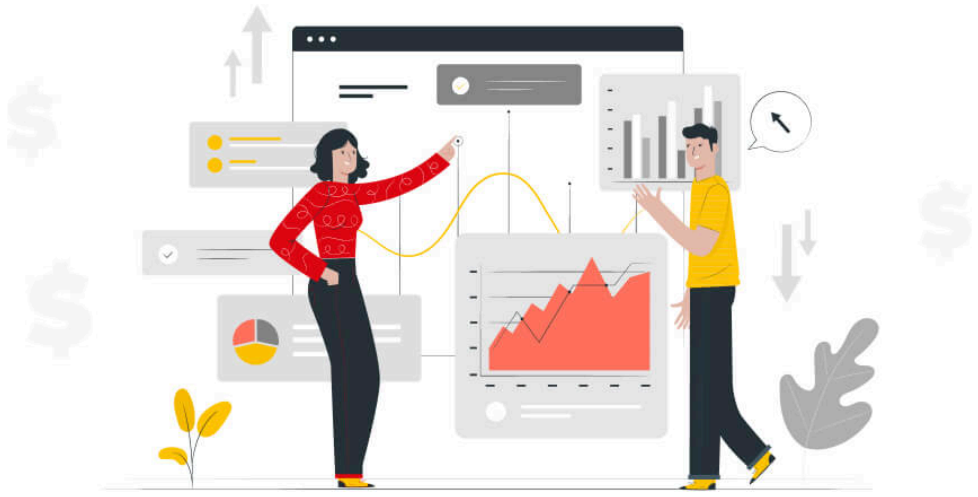
The user instructions may include:

- Start guide of the product
- Instructions on how to install the software
- The troubleshooting guide
- Video tutorials
- FAQs
- Support Portals
- Description of the functionality of the product
- System admin guide

Well-written user documentation can save a significant amount of time on training and adaptation of the user to the program, and reduce the number of mistakes that can increase the system's economic efficiency.

DIGITAL METRICS TO MEASURE THE PRODUCT SUCCESS

To know if a **digital product is successful (https://upplabs.com/blog/tracking-and-improving-digital-product-metrics-best-practices/)**, we need to have a qualitative or quantitative indicator that reflects a particular characteristic and product success level. Quantitative metrics are easier to track and are used more frequently. Based on these numbers, you can always get an idea about what is happening in general: whether users need such a product, how much they like it, whether it solves their problem.



Digital metrics. By Upplabs

There is no need to create your own tool for measuring your digital product metrics. All the metrics that are vital for your business success can be counted and analyzed using already existing free or paid online tools.

Some of the tools UppLabs use for measuring products' performance are:

- Google Analytics
- Matomo
- Woopra
- Mixpanel
- StatCounter
- Hotjar
- Kissmetrics

and others.

Whatever product you aim to build, UppLabs will help you choose the best analytic service to track crucial metrics. We provide numerous integrations (https://upplabs.com/services/device-and-third-party-software-integration/)within your project development: from APIs to wearable devices that collect and analyze user data.

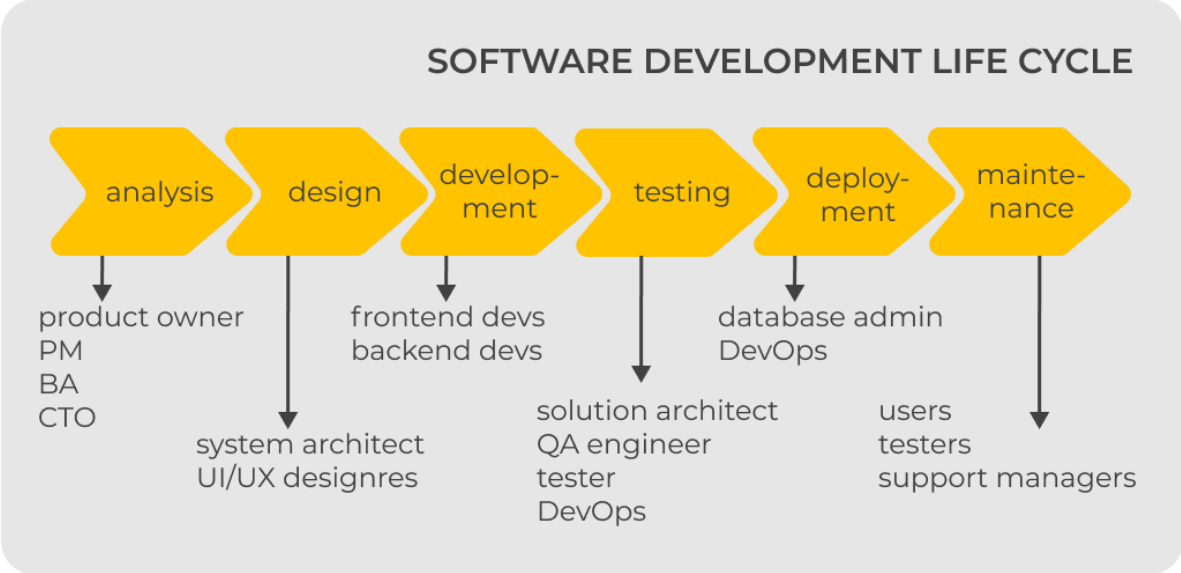
The process of developing a software product seems complicated only at first glance. Yes, you will have to make many important decisions and constantly return to the previous stages. But in the end, you will receive a well-deserved reward of a quality product and grateful users.

Check UppLabs portfolio (https://upplabs.com/portfolio/)

SUMMARY

This software development roadmap depends on the SDLC model you choose. However, the planning, design, development, and testing phases will be common for main projects. Sometimes, there can be several roadmaps for one project, developed for end-users, clients, or managers. It can be efficient for some projects, but more often – it turns into a mess and the need to maintain the relevance of several documents. Remember that the main purpose of the project roadmap is to define the objectives of the project before you schedule a startup meeting and draw up a project plan.

With an effective project roadmap, you can give your team an idea of where you are going, how you are planning to reach your goal, and who will be traveling with you. A project roadmap is the beginning of all other project planning, and creating a strong roadmap is a great way to start a successful project.



SDLC by UppLabs

HOW UPPLABS CAN HELP

Our software development company works end-to-end with the clients, discussing all possible scenarios and questions. Starting from strategy to digital, we bring transformational outcomes. It is UppLabs' task to show you the opportunities, needs, and threats.

Our assurance software service provider includes:

1. Designing and applying appropriate project management standards
2. Planning and monitoring the project (timelines and budget)
3. Managing project risks
4. Ensuring customer satisfaction
5. Organizing and motivating a project team
6. Creating detailed, comprehensive, and well-structured technical documentation
7. Estimating, prioritizing, planning, and coordinating testing activities
8. Developing and applying development and testing processes for new and existing products to meet client needs
9. Discovery session (https://docs.google.com/forms/d/e/1FAIpQLSc_fqLcTRHZDFUf7d4arUFC2bXIgoEATSoSxfXExpAnSgEjCw/viewform)
10. CI/CD (https://upplabs.com/blog/software-delivery-practices-guide-deployment-and-development/) (Continuous Integration and Continuous Delivery)

You can always book a call with Upplabs (https://upplabs.com/contacts/upplabs-calendly/) and delegate the task with a value proposition to us. Every day, we implement complex and challenging web and mobile projects. We have accumulated extensive experience in subtleties and nuances, which will undoubtedly help improve any project. We try to help the customer realize the best products because their success is our success too.
If you have any questions – UppLabs is always here to help you (https://upplabs.com/contact-us/)! We hope to win your business and build a long-term relationship with you!

LET'S HAVE A TALK
HEADS UPP!
WE KNOW WHAT TO DO!

Talk to us (https://upplabs.com/contact-us/)

READY TO DISCUSS YOUR IDEA?

FULL NAME

PHONE NUMBER

E-MAIL

COUNTRY

MESSAGE

SEND MESSAGE

(https://upplabs.com)

PHONE :

The US: +1 813.556.6452 (PA)
(tel:+18135566452)

EMAIL :

hello@upplabs.com (mailto:hello@upplabs.com?subject=Cooperation email to UPPLABS LLC) (mailto:hello@upplabs.com?subject=Cooperation email to UPPLABS LLC)

HEADQUATERS

USA, Cyprus

R&D OFFICE

Ukraine

TECHNOLOGY STACK:

.net (https://upplabs.com/skills/net/)

Python (https://upplabs.com/skills/python/)

Node.js (https://upplabs.com/skills/node-js/)

Vue.js (https://upplabs.com/skills/vue-js/)

React.js (https://upplabs.com/skills/react-js/)

Java (https://upplabs.com/skills/java/)

React Native (https://upplabs.com/skills/react-native/)

Swift (https://upplabs.com/skills/swift/)

Kotlin (https://upplabs.com/skills/kotlin/)

SERVICES:

Fintech solutions development (https://upplabs.com/services/fintech-solutions-development/)

Web and mobile app development (https://upplabs.com/services/web-and-mobile-app-development/)

Extended Team (https://extendedteam.upplabs.com/)

Legacy Rebuild (https://upplabs.com/services/legacy-rebuild/)

Device and software integration (https://upplabs.com/services/device-and-third-party-software-integration/)

Discovery phase (https://upplabs.com/services/discovery-phase/)

Quality Engineering & Testing (https://upplabs.com/services/quality-engineering-testing/)

EXPERTISE

Fintech (https://upplabs.com/expertise/fintech/)

Healthcare (https://upplabs.com/expertise/healthcare/)

Real Estate (https://upplabs.com/expertise/real-estate/)

WHY US? (https://upplabs.com/why-should-you-work-with-upplabs/)

REVIEWED BY

(https://www.pangea.ai/vendors/upplabs)

in(https://... 7YN9dvppOBjR7sOEgxA)
chat) calendly/)