Spesifikasi Kebutuhan Perangkat Lunak

untuk

<NAME>

Versi 1.0

Disiapkan oleh <author>

<organization>

30 Maret 2022

Daftar Isi

Daftar Isi ii

Daftar Revisi ii

1. Pendahuluan 1

1.1 Tujuan 1

1.2 Definisi, Akronim, dan Singkatan 1

1.3 Cakupan Proyek 1

2. Deskripsi Proyek 1

2.1 Gambaran Umum Produk 1

2.2 Fitur Produk 1

2.3 Diagram *Use Case* 2

2.4 Diagram Arsitektur Sistem 2

2.5 *Entity Relationship Diagram* 2

2.6 Pengguna 2

2.7 Lingkungan Operasi 2

2.8 Batasan Proyek 2

2.9 Dokumen Pengguna 3

2.10 Asumsi dan Dependensi 3

3. Antarmuka dan Kebutuhan Fungsional 3

3.1 Antarmuka (Wireframe) 3

3.2 Kebutuhan Fungsional 4

4. Kebutuhan Non-Fungsional 4

4.1 Performance Requirements 4

4.2 Safety Requirements 4

4.3 Security Requirements 4

4.4 Software Quality Attributes 4

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: Issues List 5

Daftar Revisi

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama** | **Tanggal** | **Alasan Revisi** | **Versi** |
|  |  |  |  |
|  |  |  |  |

# Pendahuluan

## Tujuan

Tujuan dari dokumen ini adalah untuk memberikan gambaran tentang kebutuhan proyek, kriteria penerimaan kebutuhan proyek, akses kontrol pengguna, batasan, dan antarmuka antar komponen. Dokumen ini harus ditinjau dan diusulkan kepada klien untuk persetujuan. Dokumen yang disetujui akan berfungsi sebagai dasar tetap selama proyek.

## Definisi, Akronim, dan Singkatan

|  |  |
| --- | --- |
| **Istilah** | **Definisi** |
| **SI** | Sistem Informasi |
| **KPI** | *Key Performance Index/* Indeks Kinerja Utama |
| **Admin/Administrator** | Orang yang diberi wewenang untuk mengelola dan mengendalikan system |
| **DB** | *Database/*Basis Data |
| **DESC** | Deskripsi |
| **ACC** | *Acceptance Criteria*/ Kriteria persetujuan |

## Cakupan Proyek

<Proyek> ini bertujuan untuk membuat sebuah aplikasi web dengan fungsi mengelola KPI dari pengguna dimana pengguna dapat melaporkan pekerjaan yang dilakukan, mengunggah bukti pekerjaan, dan melihat total perolehan nilai yang sudah dikumpulkan baik dari pengguna itu sendiri maupun dari bawahannya. Pengguna juga dapat menambah, mengubah dan menghapus aktivitas-aktivitas KPI yang ada. Dengan <proyek> ini, pengguna tidak harus menghitung skor KPI dan melaporkannya secara fisik.

# Deskripsi Proyek

## Fitur Produk

<Summarize the major features the product contains or the significant functions that it performs or lets the user perform. Details will be provided in Section 3, so only a high level summary is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or a class diagram, is often effective.>

## Diagram *Use Case*

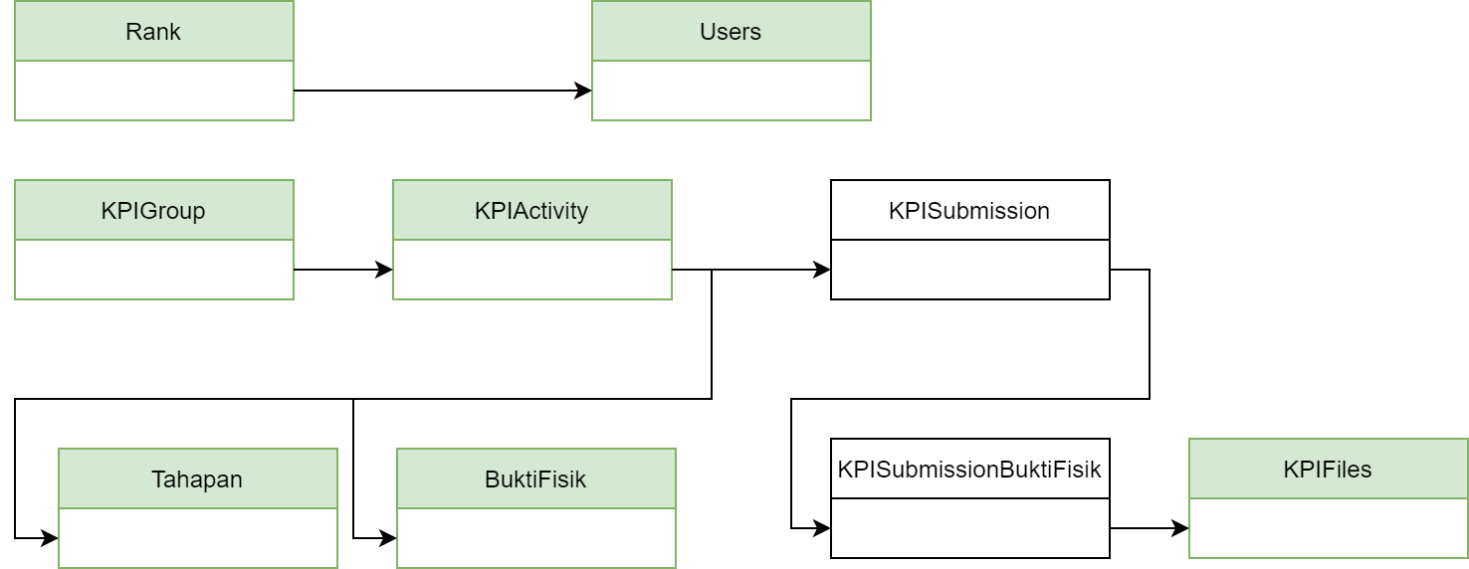
<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

## Diagram Arsitektur Sistem

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

## 

## *Entity Relationship Diagram*



<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

## Diagram Kelas

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the favored user classes from those who are less important to satisfy.>

## Pengguna

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the favored user classes from those who are less important to satisfy.>

## Lingkungan Operasi

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

## Batasan Proyek

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>

## Dokumen Pengguna

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

## Asumsi dan Dependensi

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

# Antarmuka dan Kebutuhan Fungsional

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## Antarmuka (Wireframe)

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

3.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

3.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

3.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:

REQ-2:

## Kebutuhan Fungsional

# Kebutuhan Non-Fungsional

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: Issues List

< This is a dynamic list of the open requirements issues that remain to be resolved, including TBDs, pending decisions, information that is needed, conflicts awaiting resolution, and the like.>