Software Requirements Specification (SRS) for

Encrypted Chatting Project

*Versi Dasar 0.1*

*Diterbitkan pada : 3 Augustus 2015*

Diterbitkan oleh : Stefanus Denny Yahya.

Diterbitkan untuk : UPH

**Change History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Changes** |
| 0.1 | July 5, 2015 | Stefanus Denny Yahya | Added AES Algorithm |
| 1.0 | July 5, 2015 | Stefanus Denny Yahya | Initial version |
| 1.1 | July 27, 2015 | Stefanus Denny Yahya | Replaced AES with Caesar Chiper Algorithm |
| 1.2 | July 27, 2015 | Stefanus Denny Yahya | Added Integration Testing |
| 1.3 | July 30, 2015 | Stefanus Denny Yahya | Added Unit Testing and Data Collection |

**Document Approval**

The following Software Requirements Specification has been accepted and approved by the following :

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Title** | **Date** | **Signature** |
| Stefanus Denny Yahya | Project Manager | August 3, 2015 |  |

**Table of Contents**

**List of Figures**

**List of Tables**

**Chapter 1**

**INRODUCTION**

* 1. **Purpose**

*We want to know what are the requirements that we should fulfill in developing this software.*

* 1. **Scope**
  2. *Explain what the software product will (and will not do) upon completion*
  3. *Describe the application of the software being specified. As a portion of this, it should :*
     1. *Describe all relevant benefits, objectives, and goals as precisely as possible. For example, to say that one goal is to provide effective reporting capabilities is not as good as saying parameter-driven, user-definable reports with an on-line entry of user parameters.*
     2. *Be consistent with similar statements in higher-level specifications (if they exist). What is the scope of this software product? Does it include the hardware parts?*
  4. **Definitions, Acronyms, and Abbreviations**

*You can use this section to list all phrases, terms, and acronyms that are required to properly understand this SRS. This information may refer to other documents that belong to your documentation.*

* 1. **References**

*This subsection is used to :*

1. *Provide a complete list of all documents referenced elsewhere in the SRS, or in a separate, specified document.*
2. *Identify each document by title, report number (if applicable) date, and publishing organization.*
3. *Specify the sources from which the references can be obtained.*

*This information may refer to other documents.*

* 1. **Overview**

*This part is used to describe how your SRS will be organized.*

**Chapter 2**

**GENERAL DESCRIPTION**

1. **Product Perspective**

*This subsection of the SRS puts the product into perspective with other related products or projects, whether it is independent and totally self-contained or not. If the product is a component of a larger system, then describe the functions of each component of the larger system and identify the interfaces, overview the principal external interfaces of this product, and overview the hardware and any peripheral equipment to be used. Give a block diagram that shows the major components of the product, interconnections, and external interfaces.*

1. **Product Functions**

*This subsection of the SRS should provide a summary of the functions that the software will perform. Present this part in a most readable way that is understandable by users.*

1. **User Characteristics**

*This subsection of the SRS should describe those general characteristics of the eventual users of the product that will affect the specific requirements.*

1. **General Constraints**

*This subsection of the SRS should provide a general description of any other items that will limit the developer’s options for designing the system. Some of them are : regulatory policies, hardware limitations, interfaces to other applications, parallel operation, audit functions, control functions, criticality of the application, safety and security considerations, etc.*

1. **Assumptions and Dependencies**

*This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.*

**Chapter 3**

**SPECIFIC REQUIREMENTS**

*This will be the largest and most important section of the SRS. The customer requirements will be embodied within Chapter 2, but this section will give the requirements that are used to guide the project’s software design, implementation, and testing.*

*Each requirement in this section should be :*

* *Correct*
* *Traceable (both forward and backward to prior/future artifacts)*
* *Unambiguous*
* *Verifiable (testable)*
* *Prioritized (with respect to importance and/or stability)*
* *Complete*
* *Consistent*
* *Uniquely identifiable (usually via numbering, like 3.4.5.6)*

*Attention should be paid to the carefully organize the requirements presented in this section so that they may easily accessed and understood. Furthermore, this SRS is not the software design document, therefore one should avoid the tendency to over-constrain (and therefore design) the software project within this SRS.*

1. **External Interface Requirements**

*This part can describe about any interface that you may face. It can be user interface, like screen format, function keys, etc. It can be hardware interface, like how the software and hardware will communicate. It can also be software interface, like what OS to be used, what DBMS, or any other software packages. It can also include about communication interface, database (frequency of use, accessing capabilities, static and dynamic organization, etc).*

1. **Functional Requirements**

*Clear*

1. **Use Cases**

*Create six use cases, represented in a use case diagram. Each use case should come with a scenario (use full dressed style). Each scenario can have different numbers of activities, from which you will choose to emphasize only (1.5 x numberOfYourMembers) activities. This will determine how many activity diagrams and state chart diagrams that you’ll create.*

1. **Classes/Objects**

*Clear. You are required to create a class diagram. If necessary, you can create a list of explanation of each class.*

1. **Non-Functional Requirements**

*Clear*

1. **Design Constraints**

*In this part, you can explain about constraints that limit your project. It can be about software design constraints, like your standards for design, coding, naming, use of application package, constraints on program size, data size, etc. It can also be hardware design constraints, like specific type of hardware, its reliability, its interface, requirements for spare capacity or performance, etc.*

1. **Other Requirements**

*You can specify other requirements here. There are many requirements that you can explain here, like security, safety, environmental, reusability, training, etc.*

**Chapter 4**

**ANALYSIS MODELS**

*List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.*

1. **Sequence Diagrams**
2. **State-Transition Diagrams (STD)**

**Chapter 5**

**CHANGE MANAGEMENT PROCESS**

*Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how these changes will be approved.*

1. **Appendices**

*Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements.*

1. **Appendix 1**
2. **Appendix 2**