Software requirement specification (SRS) document template

Project name:

Date:

Version:

By:

Revision history

| Version | Author | Version Description | Date Completed |
| --- | --- | --- | --- |
| 1 | Mitchell Seng | First iteration | — |

Review history

| Approving party |  |
| --- | --- |
| Version approved |  |
| Signature date |  |

Approval history

| Reviewer |  |
| --- | --- |
| Version reviewed |  |
| Signature |  |
| Date |  |

Table of contents

1. Introduction
   1. Product scope
   2. Product value
   3. Intended audience
   4. Intended use
   5. General description
2. Functional requirements
3. External interface requirements
   1. User interface requirements
   2. Hardware interface requirements
   3. Software interface requirements
   4. Communication interface requirements
4. Non-functional requirements
   1. Security
   2. Capacity
   3. Compatibility
   4. Scalability
   5. Maintainability
   6. Usability
   7. Other non-functional requirements
5. Definitions and acronyms

Introduction   
Describe the purpose of the document.

1.1 Product scope

* The aim of this project is to create an application for a bank. The application will offer a secure place for users to do their banking. Using two factor authentication and data encryption to keep a user's information safe. Another benefit of the project is the ability to contact support from the application directly.

1.2 Product value

* The project aims to offer a secure place for users to do their banking, this security should add to the value they hold in the product. Another thing that will add value to the product is the ability to contact a help desk through the application. By not having to exit the banking application to get support the user gets a more cohesive experience.

1.3 Intended audience

* The intended audience of this product is the general public, users may vary in age and culture. The product should be able to be used by anyone to do their banking.

1.4 Intended use

* When the user first opens the application they will be prompted with a login page. After they have logged into their account they will be given various options, including edit account details, close or open an account, move money between accounts, move money into or out of the account, view a transaction history, set up recurring bills, and manage services such as checkbooks and credit cards.

1.5 General description

* The software will offer users a range of banking functions. Starting with account management users will be able to set up two types of banking accounts, personal or business. They will be able edit the account details including the account type, and any user information that may need updating. Finally they will be able to close any account they have opened. Next users will be able to make financial transactions such as depositing money into an account, withdrawing money from an account, and transferring money between different accounts within the same bank. All these transactions will be able to be viewed in an accounts transaction history. For account services a user will be able to view account information, including its current balance, recent transactions, and the account type. Request and manage checkbooks, debit, and credit cards. And set up recurring payments using direct debits. All these functions will comply with relevant and current data laws and regulations. This will be done with a robust two factor authentication system and appropriate levels of data encryption for any sensitive information. Overall the software will provide a user friendly and comprehensive banking experience.

Functional requirements

The banking applications user interface should be responsive and intuitive ensuring easy navigation and accessibility across various devices. Compliance with accessibility standards are essential to ensure users have a good experience while using the application. Another essential requirement is implementation of a scalable and modular system, this is to accommodate the potential growth and facilitate any maintenance that may need to take place.

* Cross-browser compatibility is crucial to reaching a broad user audience. In terms of graphics requirements, high-quality visuals and graphical representations of financial data are emphasized, along with the incorporation of responsive images.
* Operating system requirements mandate cross-platform compatibility, ensuring smooth operation on major systems such as Windows, macOS, and Linux, as well as compatibility with popular web browsers.
* The software must adhere to regulatory constraints, emphasizing compliance with financial regulations and data protection laws, while also prioritizing robust security measures.
* Budget and timeline constraints need careful consideration to meet development and maintenance requirements without compromising quality.
* Adherence to the chosen technology stack and compatibility with existing infrastructure constitute additional constraints for the successful implementation of the banking software.

External interface requirement

3.1 User interface requirements Describe the logic behind the interactions between the users and the software (screen layouts, style guides, etc).   
3.2 Hardware interface requirements List the supported devices the software is intended to run on, the network requirements, and the communication protocols to be used.   
3.3 Software interface requirements Include the connections between your product and other software components, including frontend/backend framework, libraries, etc.   
3.4 Communication interface requirements List any requirements for the communication programs your product will use, like emails or embedded forms.

Non-functional requirements

4.1 Security Include any privacy and data protection regulations that should be adhered to.  
4.2 Capacity Describe the current and future storage needs of your software.   
4.3 Compatibility List the minimum hardware requirements for your software.   
4.4 Reliability Calculate what the critical failure time of your product would be under normal usage.   
4.5 Scalability Calculate the highest workloads under which your software will still perform as expected.   
4.6 Maintainability Describe how continuous integration should be used to deploy features and bug fixes quickly.   
4.7 Usability Describe how easy it should be for end-users to use your software.   
4.8 Other List any additional non-functional requirements.

Definitions and acronyms