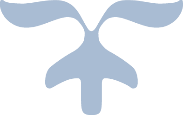


TechBook

Note Taking Application



Project submitted in the context of the course   
I3301 Software Engineering

Prepared by   
Hussein Ali Allaw - 106900

Lebanese University

Faculty of Sciences I

Department of computer Sciences

2022 - 2023

# Abstract

The chosen application/domain and the major steps done while achieving the project

# Table of Contents

[Acknowledgements 3](#_Toc90274882)

[Abstract 4](#_Toc90274883)

[Table of Contents 5](#_Toc90274884)

[Table of Figures 8](#_Toc90274885)

[Table of Tables 9](#_Toc90274886)

[How to format your text in this document 10](#_Toc90274887)

[Chapter I - Introduction 12](#_Toc90274888)

[I.1. The business domain 12](#_Toc90274889)

[I.2. About the modeled application (Purpose, Users Persona and Product Perspective) 13](#_Toc90274890)

[I.3. Analysis of the Existing Similar Apps 13](#_Toc90274891)

[I.4. Plan of the document 14](#_Toc90274892)

[Chapter II - Requirement Analysis and Specification (extension of SRS document provided by IEEE) 15](#_Toc90274893)

[II.1. Introduction 15](#_Toc90274894)

[1.1. Purpose 15](#_Toc90274895)

[1.2. Document Conventions 15](#_Toc90274896)

[1.3. Intended Audience and Reading Suggestions 15](#_Toc90274897)

[1.4. Product Scope 15](#_Toc90274898)

[1.5. References 15](#_Toc90274899)

[II.2. Overall Description 16](#_Toc90274900)

[2.1. Product Perspective 16](#_Toc90274901)

[2.2. Product Functions 16](#_Toc90274902)

[2.3. User Classes and Characteristics 16](#_Toc90274903)

[2.4. Operating Environment 16](#_Toc90274904)

[2.5. Design and Implementation Constraints 16](#_Toc90274905)

[2.6. User Documentation 16](#_Toc90274906)

[2.7. Assumptions and Dependencies 17](#_Toc90274907)

[II.3. External Interface Requirements 17](#_Toc90274908)

[3.1. User Interfaces 17](#_Toc90274909)

[3.2. Hardware Interfaces 17](#_Toc90274910)

[3.3. Software Interfaces 17](#_Toc90274911)

[3.4. Communications Interfaces 17](#_Toc90274912)

[II.4. System Features 18](#_Toc90274913)

[4.1. System Feature 1 18](#_Toc90274914)

[4.2. System Feature 2 (and so on) 18](#_Toc90274915)

[II.5. Other Nonfunctional Requirements 18](#_Toc90274916)

[5.1. Performance Requirements 18](#_Toc90274917)

[5.2. Safety Requirements 19](#_Toc90274918)

[5.3. Security Requirements 19](#_Toc90274919)

[5.4. Software Quality Attributes 19](#_Toc90274920)

[5.5. Business Rules 19](#_Toc90274921)

[II.6. Other Requirements 19](#_Toc90274922)

[II.7. Requirements Gathering 19](#_Toc90274923)

[II.8. Requirements Analysis 20](#_Toc90274924)

[8.1. Use Cases 21](#_Toc90274925)

[8.2. Use Cases Textual Description 21](#_Toc90274926)

[II.9. High-Level Design Specification 22](#_Toc90274927)

[II.10. Conclusion 23](#_Toc90274928)

[Chapter III - Application Conception 24](#_Toc90274929)

[III.1. Introduction 24](#_Toc90274930)

[III.2. UML Class Diagram 24](#_Toc90274931)

[III.3. Sequence Diagrams 24](#_Toc90274932)

[III.4. State Chart Diagram 24](#_Toc90274933)

[III.5. Conclusion 24](#_Toc90274934)

[Chapter IV - Conclusion 26](#_Toc90274935)

[IV.1. Future Considerations 26](#_Toc90274936)

[Bibliography 27](#_Toc90274937)

[Note about the bibliography [to be deleted from the final document] 27](#_Toc90274938)

# Table of Figures

[Figure 1 Predefined Styles in this Document 10](#_Toc90274939)

[Figure 2 References Ribbon 11](#_Toc90274940)

[Figure 3 Mobile banking boom in recent years 12](#_Toc90274941)

[Figure 4 Sample Sketch Mockup 20](#_Toc90274942)

[Figure 5: Example for a DTD 22](#_Toc90274943)

# Table of Tables

[Table 1 Comparison of Existing Similar Applications 13](#_Toc90274944)

1. Introduction

This chapter introduces the application being modeled and the underlying business domain. It gives the motivation by showing both, its importance and that the work to be achieved was lacking.

* 1. The business domain

Few words about it and about its potential, with ideally numbers and stats supported by valid references, for example as shown in Figure 3.

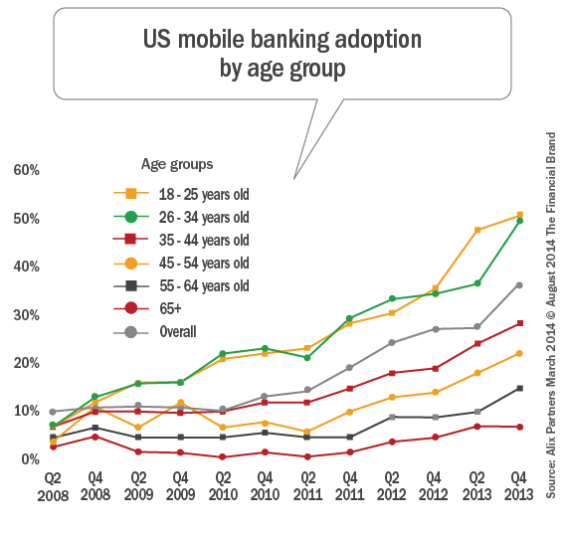


Figure Mobile banking boom in recent years

* 1. About the modeled application (Purpose, Users Persona and Product Perspective)

This section aims at emphasizing the importance of our application. Start by showing how much the underlying business really needs an application such as the one you intend to model in this document.

The application is intended to be used by… for… to do…

We will give concrete examples and situations where the app will play a main role in helping/saving clients’ time…

Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product.

* 1. Analysis of the Existing Similar Apps

Check and tell whether similar applications already exist or not. In case they exist, give few words about their features, some screenshots to showcase what they typically offer to their users.

The existing applications, presented above, can be synthesized in the following table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Feature 1 | Feature 2 | Feature 3 | Feature 4 |
| App 1 |  |  |  | X |
| App 2 | X | X |  |  |
| App 3 | X |  |  |  |
| App 4 |  | X |  | X |
| App 5 |  |  | X |  |

Table Comparison of Existing Similar Applications

As we can notice in the above table,

The **software development life cycle** that best fit your chosen business model, application nature and imagined team of developers that will be working on its implementation…

* 1. Plan of the document

In this document, we propose a specification, design and project planning for an Abc XyZ application. We start by showing … then …

1. Requirement Gathering, Analysis, and Specification (extension of SRS document provided by IEEE)
   1. Introduction
      1. Purpose

Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.

* + 1. Document Conventions

Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.

* + 1. Intended Audience and Reading Suggestions

Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.

* + 1. Product Scope

Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.

* + 1. References

List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.

* 1. Overall Description
     1. Product Perspective

Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.

* + 1. Product Functions

Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 5, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS.

* + 1. User Classes and Characteristics

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 most important user classes for this product from those who are less important to satisfy.

* + 1. Operating Environment

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.

* + 1. Design and Implementation Constraints

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).

* + 1. User Documentation

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.

* + 1. Assumptions and Dependencies

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).

* 1. External Interface Requirements
     1. User Interfaces

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.

* + 1. Hardware Interfaces

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.

* + 1. Software Interfaces

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.

* + 1. Communications Interfaces

Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.

* 1. Requirements Gathering

State your strategy to gather the requirements: questionnaire, iterations on several prototypes, Interviews, etc… **Explain the sequel of the information and requirements gathering phase.**

Give samples of sketches and/or prototype GUI screenshots, for example:



Figure Sample Sketch Mockup

Famous Mockup Tools:

* <http://www.balsamiq.com>
* <https://moqups.com>
* <http://www.axure.com>
* <https://www.adobe.com/products/xd.html>
  1. System Features

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.

* + 1. System Feature 1

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

* 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).

* 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.

* 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:

* + 1. System Feature 2 (and so on)
  1. Other Nonfunctional Requirements
     1. 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.

* + 1. 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.

* + 1. 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.

* + 1. 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.

* + 1. Business Rules

List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.

* 1. Other Requirements

Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.

* 1. Requirements Analysis

List the collected and identified requirements. For each, state the category based on the Audience-Oriented Requirements. Criticize the given set of required features**, especially if you do not agree with the client, or if you are unable to justify them**, or unable to imagine whether you will succeed in fulfilling 2 or more conflicting requirements.

**Based on MOSCOW Method, prioritize your requirements.** For every non-functional requirement, **explain how to verify and validate it with the client**.

* + 1. Use Cases

To draw your UML diagrams, use a computerized tool such as Modelio[[1]](#footnote-1) or StarUML[[2]](#footnote-2) or Visual Pradigm[[3]](#footnote-3). **Mandatory**… Give the whole UC diagram then explain and justify all the association in it (extends, includes and inheritance).

* + 1. Use Cases Textual Description

Good to have it for the couple of main functionalities… optional for the rest. The following is a sample use case detailed textual description:

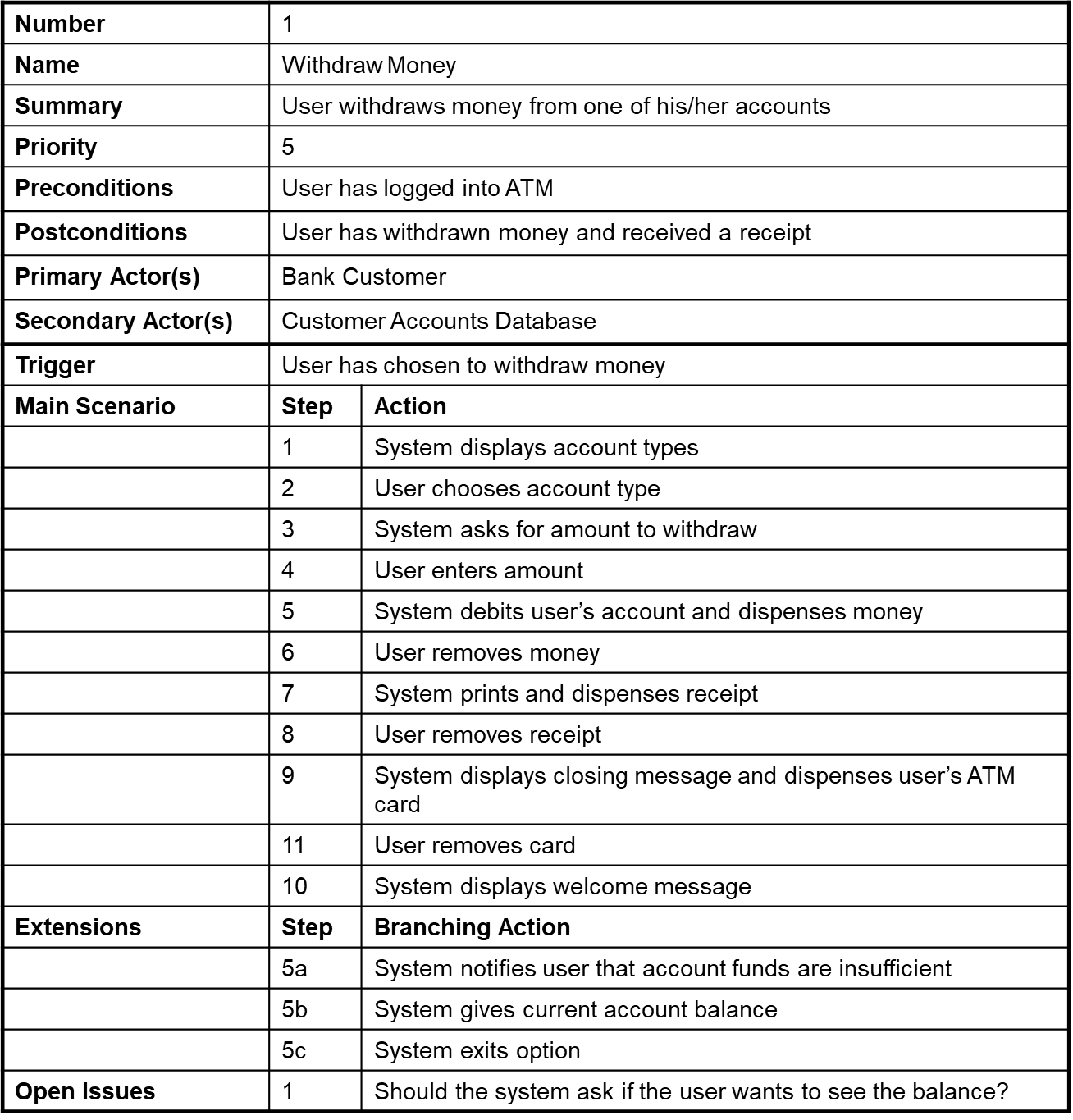


Figure : Example for a DTD

* 1. High-Level Design Specification

Exactly what you should **specify** in the high‐level design varies somewhat, but some things are constant for most projects.

* Security
* Hardware
* User Interface
* Internal Interfaces
* External Interfaces
* Architecture
* Reports
* Database
  1. Conclusion

This chapter presented…

Difficulties encountered in gathering the data, if any, and the followed approach to overcome them (ask the client for more details, check and get inspired by existing similar apps, leaving the questions open till being answered in an agile reactive way later on during the development phase…)

1. Application Conception
   1. Introduction

In the previous chapters, we have talked about the features that should be offered by our application... This reveals that the following entities are implied in the process… In this chapter we give, using UML class diagrams, the conceptual model that clarifies these entities, their underlying handled data, roles in the application and associations with each other.

* 1. Database

Database is “a structured set of data held in a computer, especially one that is accessible in various ways.” There are various types of databases, each providing different functionality to their users. In this section, you should present your DB type/structure needed for you project.

* 1. UML Class Diagram

The whole picture can be cut into several smaller diagrams, each focusing on a specific set of interrelated entities and concepts.

Every diagram should be well annotated and explained in the sequel. Every choice made (association, cardinality, abstraction, etc.) and every used symbol should be justified and well explained.

* 1. Sequence Diagrams

Can be given for the main scenarios to show the sequence of functions calls between **objects and instances of the various proposed classes**.

* 1. State Chart Diagram

Opt for state chart diagrams only if one (or more) of your key and central entities change state as the process progresses and when different use cases are executed, and you want to capture this crucial business logic.

* 1. Conclusion

This chapter presented our application design. First, we introduced…

After that, we presented...

In the next chapter, we make a complete feasibility study and draw out the plan of our project realization, risk management and client support and satisfaction strategy...

1. Conclusion

About the whole project…

Discussion of the lived experience and the obtained results.

* 1. Future Considerations

Pointers toward future work, especially if you are really willing to implement the proposed application.

## Bibliography

|  |  |
| --- | --- |
| [1] | Summerville, Software Engineering, Pearson, 2014.  [2] Beginning Software Engineering, Rod Stephens. John Wiley & Sons, Inc. |
|  |  |

## Note about the bibliography [to be deleted from the final document]

The bibliography should only contain valid references, i.e. books, journal articles and companies’ white papers. URLs, such as Wikipedia, forums, normal blogs and web pages are not valid references. In case you used them in the document, then put them in footnotes.

1. <https://www.modelio.org/> [↑](#footnote-ref-1)
2. <http://staruml.io/> [↑](#footnote-ref-2)
3. <https://www.visual-paradigm.com/> [↑](#footnote-ref-3)