Contents

1 Vision

1.1 Introduction

Our goal is to make an interactive document sharing system, Slice of Pie, which allows multiple users to share and edit documents both online and offline.

1.2 Problem statement

Sharing and editing documents can be cumbersome.
Sending a document back and forth between multiple users can lead to a lot of errors. Users can overwrite what another user has done, and if they aren't all using the same text editing system this can lead to formatting issues in the document.

1.3 Summary of system features

1. Multiple users must be able to share and edit documents online.
2. Synchronization for offline usage.
3. Merging of documents.
4. History. Which allows the user to see all recent changes made to the document.
5. Documents can be categorized into folders or projects in order to get a better overview when working on a larger project with multiple files.

2 Use cases

1. UC1: Create new document
2. UC2: Edit document
3. UC3: Delete document
4. UC4: Merging documents (resolve conflict)
5. UC5: Offline sync
6. UC6: New folder
7. UC7: New project
8. UC8: Find old version of document

Use case UC1: Create new document
Scope: Slice-of-pie application
Level: User goal
Primary actor: Regular user
Stakeholders and Interests:
- Regular user: Wants to create a new document without any complications.
Preconditions: none.
Postconditions: The document must be created

Basic flow:

1. The user needs a document that can be shared with others.
2. The user chooses a client program (web or desktop client).
3. The user presses the "New Document" button.
4. The client presents a dialogue box.
5. The user types information about the document and press OK.
6. The client sends a request to the server and the document gets stored by the server.
7. The client displays the document for the user.

Extensions:

6. The server is not responding

    (a) The document is saved locally.
    (b) The client requests the server in regular intervals to see if it has recovered.

Special requirements:
- The document should be displayed properly on any devices such as smart phones and tablets.

Use case UC2: Edit document
Scope: Slice-of-pie application
Level: User goal
Primary actor: Regular user
Preconditions: A document must have been created
Postconditions: The document must be edited the way the user desires and the change must be recorded by the server too

Basic flow:

1. The user opens a document.
2. The user edits the content of the document.
3. The client sends a request to the server.
4. The document is changed on the server.

Extensions:

3. The server is not responding

    (a) The document is saved locally.
    (b) The client requests the server in regular intervals to see if it has recovered.

Special requirements:
- The document should be displayed properly on any devices such as smart phones and tablets.

Use case UC3: Delete document
Scope:
Level:
Primary actor:
Preconditions:
Postconditions:

Basic flow: Extensions: Special requirements:

Use case UC4: Merging documents (Resolve conflict)
Scope:
Level:
Primary actor:
Preconditions:
Postconditions:

Basic flow:
Extensions:
Special requirements:

Use case UC5: Offline synchronization
Scope:
Level:
Primary actor:
Preconditions:
Postconditions:

Basic flow:
Extensions:
Special requirements:

Use case UC6: Create new folder
Scope:
Level:
Primary actor:
Preconditions:
Postconditions:

Basic flow:
Extensions:
Special requirements:

Use case UC7: Create new project
Scope:
Level:
Primary actor:
Preconditions:
Postconditions:

Basic flow:
Extensions:
Special requirements:

Use case UC8: View old version of document
Scope:
Level:
Primary actor:
Preconditions:
Postconditions:

Basic flow:
Extensions:
Special requirements:

3 Glossary

- **Response Time:** The time it takes for the system to respond to a request from the user.

4 Supplementary Specification (FURPS+)

- Functionality
  - The system must be able to store documents.
  - The system must support multiple users.
- Usability
  - The system must have a clean and simple user interface.
  - Users must be able to use the application without training.
  - System features must be simple and straight forward. No complicated work flows.
  - The overall system must be easy to use. 8 out of 10 users must be able to use the system without any training.

- Reliability
  - The system must be able to restore it's state in case of a server failure.
- Performance
  - The system must be fast and responsive. A request to the system can't take longer than 3 seconds. (not counting the external internet connection)
- Supportability

5 Domain Model

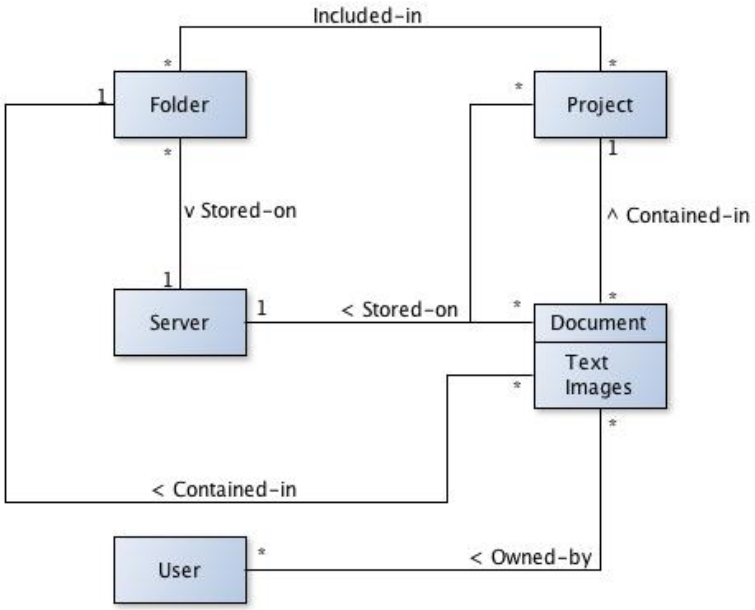

Figure 1: Domain Model

6 Logical Architecture

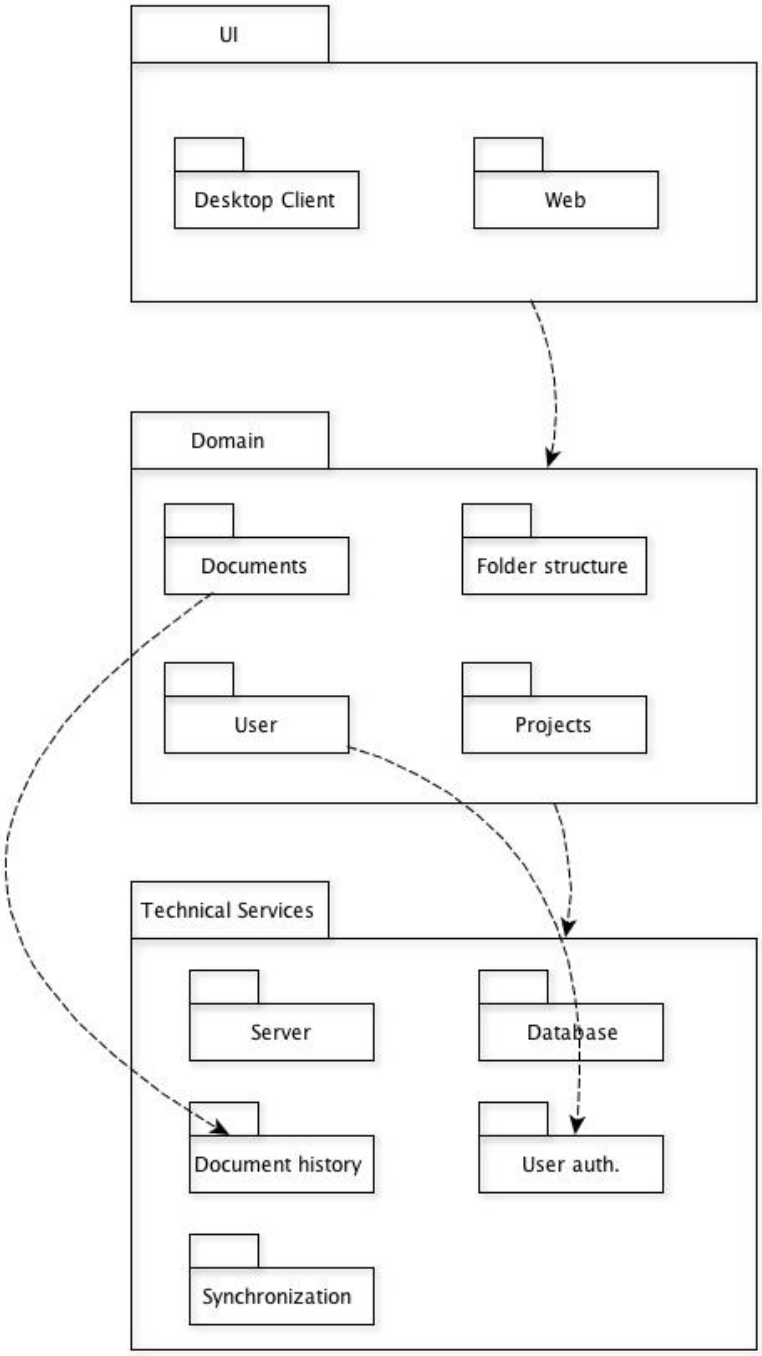

Figure 2: Logical Architecture

7 Revision Table

Table 1: Revision Table

Revision	Changes	Section
21-11-12	Created Vision	1
21-11-12	Created Use cases	2
21-11-12	Created Glossary	3
21-11-12	Created Supplementary Specifications	4
21-11-12	Created Domain Model diagram	5
21-11-12	Created Logical Architecture diagram	6