****

**LocAdoc**

**Database Design Document (DDD)**

**Version 0.1**

**Review Draft**

**Prepared by:**

**Abhi Jay Krishnan**

**Kim Hyeocheol**

**Rivaldo Erawan**

**Durrah Afshan**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1 Introduction 3

1.1 Document Objectives 3

1.2 Intended Audiences 4

1.3 References 4

1.4 Database Overview 4

2 Detailed Database Design 4

2.1 DynamoDB design (NoSQL database) 5

2.1.1 Object Diagram 6

2.1.2 Data dictionary 6

2.1.2.1 Data dictionary for Element: User 6

2.1.2.2 Data dictionary for Element: Password 7

2.1.2.3 Data dictionary for Element: File 7

2.1.2.4 Data dictionary for Element: Area 8

2.1.2.5 Data dictionary for Element: AdminArea (Stores the administration area) 8

2.2 SQLite database design (Relational database) 9

2.2.1 Conceptual diagram 9

3 Appendix 1 – XML Schema 10

# Introduction

The section introduces the Database Design Document (DDD) for LocAdoc to its readers.

## Document Objectives

This DDD for the LocAdoc software has the following objectives:

* Describe the design of a DynamoDB and SQLite database, that is, a collection of related data stored in one or more computerized files in a manner that can be accessed by users or computer programs via a database management system (DBMS). It can also describe the software units used to access or manipulate the data.
* To serve as the basis for implementing the database. It provides the acquirer visibility into the design and provides information needed for software support.

## Intended Audiences

This DDD is intended for the following audiences:

* Technical reviewers, Supervisor and UOW staff who must evaluate the quality of this document.
* LocAdoc developers including:

Architects, whose overall architecture must meet the requirements specified in this document.

Designers, whose design must meet the requirements specified in this document.

Programmers, whose software must implement the requirements specified in this document.

Testers, whose test cases must validate the requirements specified in this document.

## References

This DDD refers to the following references:

* Software requirement specification: SRS\_LocAdoc.docx
* Project Proposal: Project\_Proposal\_SS173D\_V1.docx

# Detailed Database Design

This section describes the actual design of different databases at varying levels of abstraction. A subsection for each of conceptual, internal, logical and physical levels.

## DynamoDB design (NoSQL database)

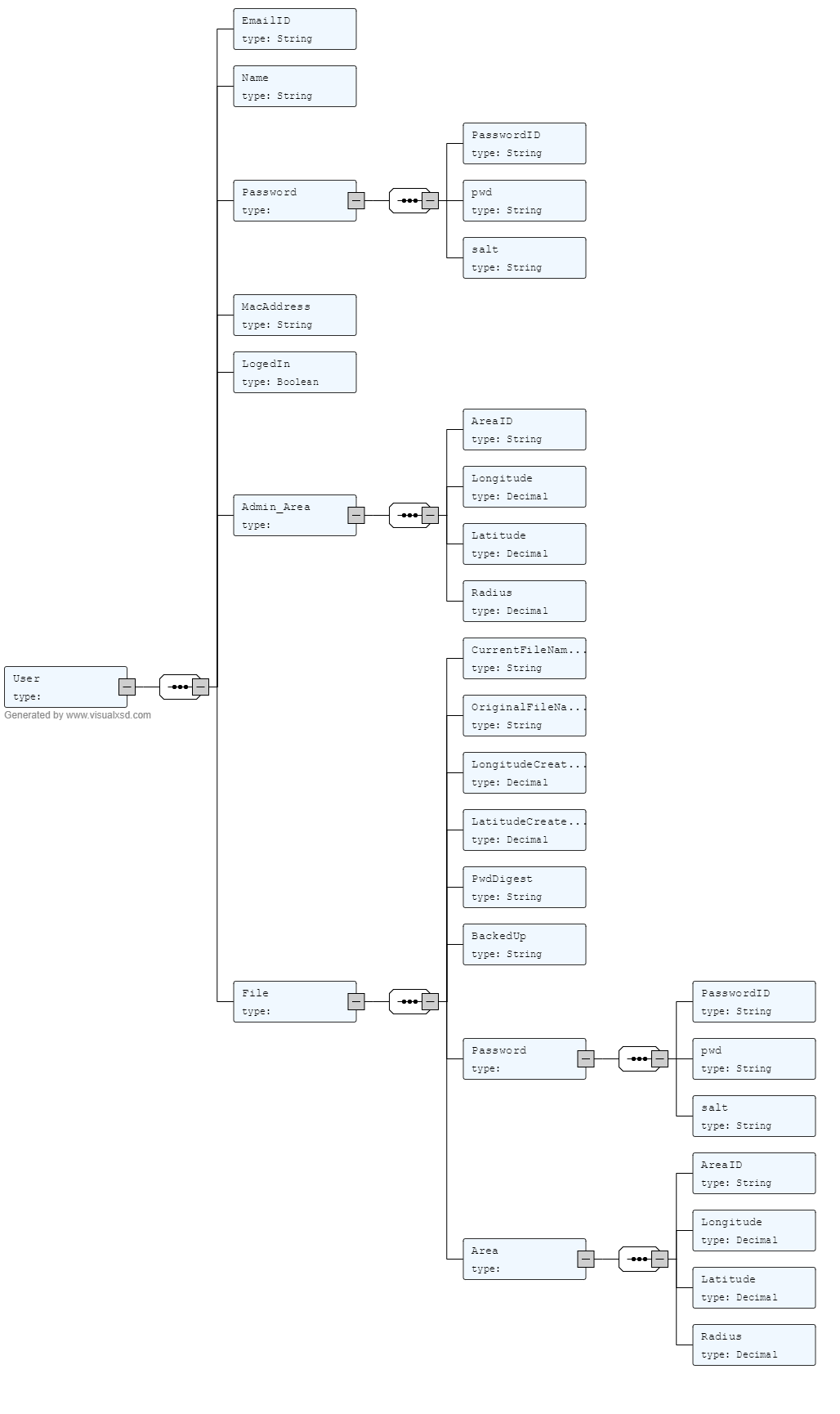


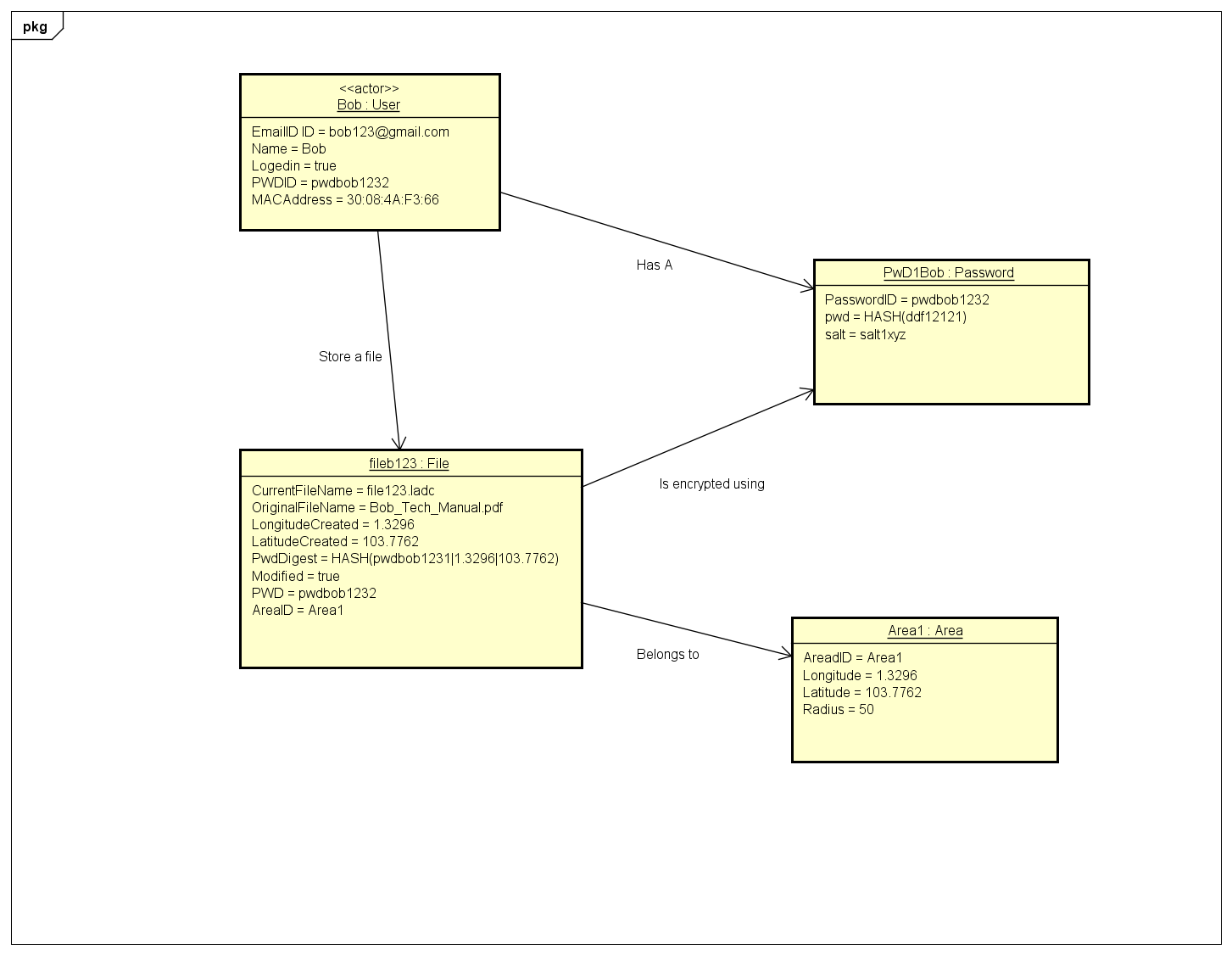
Figure 1: Database design

The diagram given above visualize the NoSQL database. This design was developed after creating a XML schema (Appendix 1) and using an online converter. [2]

The user element will hold all the user details which be mapped to there to his password and files. He will also have an option to set an user administration area.

### Object Diagram

Here is an object diagram to show a given instance of the database.



### Data dictionary

#### Data dictionary for Element: User

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Constrain | Description |
| Email ID (primary key) | string | Min :1, Max:1 | Email ID of the user |
| Name | String |  | Name of the user |
| Password (Foreign Key) | Password | Min :1, Max:1 | The password of the user |
| LogedIn | Boolean |  | Used to flag if the person is currently logged in a devise. So the second login can be detected. |
| MacAddress | String |  | Used to store mac address of the phone used by the user |

#### Data dictionary for Element: Password

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Constrain | Description |
| Password ID (primary key) | string | Min :1, Max:1 | ID to identify the password |
| Password | String |  | Hashed Password |
| Salt | String |  | Salt to prevent repeated keys being generated for encryption due to similar password. |

#### Data dictionary for Element: File

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Constrain | Description |
| CurrentFileName (primary key) | string | Min :1, Max:1 | A new name for the file assigned by the application |
| OriginalFileName | String |  | The original file name assigned by user. |
| LongitudeCreated | Decimal | Min :1, Max:1 | The system stores the longitudinal data where the file was created. |
| LatitudeCreated | Decimal | Min :1, Max:1 | The system stores the latitudinal data where the file was created. |
| PwdDigest | String |  | The password digest that is created by hashing password, salt, longitude and latitude. |
| BackedUP | Boolean |  | This a variable to make sure if the data has been backed up or new. |
| Password | Password |  | Password that was used to encrypt the file (password ID) |
| Area | Area |  | The area where the file has been grouped in (Area ID) |

#### Data dictionary for Element: Area

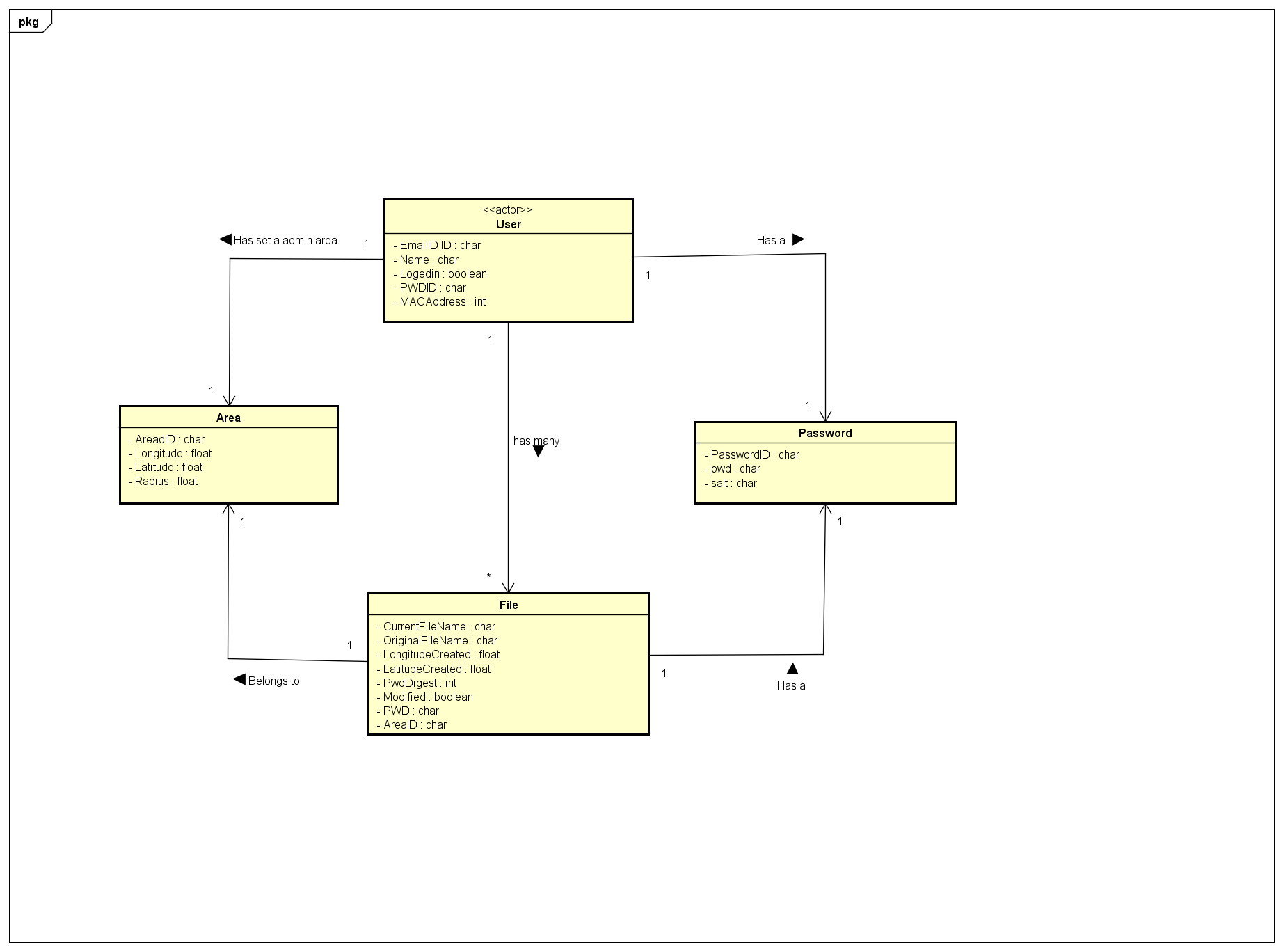
|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Constrain | Description |
| Area ID (primary key) | string | Min :1, Max:1 | ID to identify the area |
| Longitude | Decimal |  | The Longitude of the first file that was created in this area |
| Latitude | Decimal |  | The Latitude of the first file that was created in this area. |
| Radius | Decimal |  | The radius around the point where the first file was created. |

#### Data dictionary for Element: AdminArea (Stores the administration area)

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Constrain | Description |
| Area ID (primary key) | string | Min :1, Max:1 | ID to identify the area |
| Longitude | Decimal |  | The Longitude of the first file that was created in this area |
| Latitude | Decimal |  | The Latitude of the first file that was created in this area. |
| Radius | Decimal |  | The radius around the point where the first file was created. |

## SQLite database design (Relational database)

### Conceptual diagram



This diagram displays the conceptual model of the SQLite database. This database will be created after the it has imported the user’s data. The user tables will have the user’s details who is currently logged in. The user will have a password and an admin area where he can make changes to his account such as changing password (optional up to user to set it up). Each user will have zero or more files.

# Appendix 1 – XML Schema

This XML schema was created to check if the schema was well formed.



