PINWALL - DATA MODEL & API

|  |  |
| --- | --- |
| Version | 0.1 |

Purpose:

This document provides detailed information on data model intended to be implemented and API designs for PINWALL app.

Database:

The data model is designed to be compatible with MongoDB.**MongoDB** (from *hu****mongo****us*) is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [document-oriented database](https://en.wikipedia.org/wiki/Document-oriented_database). Classified as a [NoSQL](https://en.wikipedia.org/wiki/NoSQL) database, MongoDB eschews the traditional table-based [relational database](https://en.wikipedia.org/wiki/Relational_database) structure in favor of [JSON](https://en.wikipedia.org/wiki/JSON)-like documents with dynamic [schemas](https://en.wikipedia.org/wiki/Database_schema) (MongoDB calls the format [BSON](https://en.wikipedia.org/wiki/BSON)), making the integration of data in certain types of applications easier and faster. Released under a combination of the [GNU Affero General Public License](https://en.wikipedia.org/wiki/GNU_Affero_General_Public_License) and the [Apache License](https://en.wikipedia.org/wiki/Apache_License), MongoDB is [free and open-source software](https://en.wikipedia.org/wiki/Free_and_open_source_software).

The reasons to opt MongoDB are

* Schemaless
* Deep query ability
* Faster querying

If everybody is not in agreement using Mongo DB, then the designed data model can be very well fit into traditional RDBMS database like MySQL.

MongoDB Primer:

|  |  |
| --- | --- |
| RDBMS | MongoDB |
| Database | Database |
| Table | Collection |
| Row | Document |
| Column | Field |

Database Name :PinWall

Collections (Tables in RDBMS)

1. Owners
2. Notes

Owners Document:

Every registered user will get a record in this collection and this field contains their name

|  |  |  |
| --- | --- | --- |
| Field Name | Data Type | Description |
| ownerName | String | Name of registered user |
| favorites | List | Array of Note Collection Objects |

*{*

*“ownerName” : “XXXXXX”,*

*“favorites” : [ ObjectId(note1),*

*ObjectId(note2)],*

*}*

Notes Document:

Every note posted will get an entry in this collections

|  |  |  |
| --- | --- | --- |
| Field Name | Data Type | Description |
| owner | ObjectId | Reference to Owner Collection |
| noteType | String | Note Type (Standard, Sponsored) |
| text | String | String to be printed in note |
| textColor | String | Color of the text in note |
| fontSize | Float | Size of the text font |
| font | String | Font of the text |

*{*

*“owner” : ObjectId(“owner”),*

*“noteType” : “noteType”,*

*“textColor” : “color”,*

*“fontSize” : 20.0,*

*“font” : “Arial”*

*}*

APIs

To be designed as REST APIs that can fetch data from database and feed data to app. Idea is to implement the API as Python Flask Application. Any other technology will do but prefer the response to be in JSON rather than XML

/api/notes/all - Fetch all the notes from DB

Example:

*{*

*"133453" : [*

*{*

*"noteType" : "Free",*

*"noteFont" : "Verdana",*

*"textColor" : "Black",*

*"textSize" : "19"*

*},*

*{*

*"noteType" : "Free",*

*"noteFont" : "Verdana",*

*"textColor" : "Black",*

*"textSize" : "19"*

*},*

*],*

*"133454" : [*

*{*

*"noteType" : "Free",*

*"noteFont" : "Verdana",*

*"textColor" : "Black",*

*"textSize" : "19"*

*},*

*{*

*"noteType" : "Free",*

*"noteFont" : "Verdana",*

*"textColor" : "Black",*

*"textSize" : "19"*

*},*

*]*

*}*

/api/notes/<ownerid> -Fetches notes for a specific owner

Example:

*{*

*"133453" : [*

*{*

*"noteType" : "Free",*

*"noteFont" : "Verdana",*

*"textColor" : "Black",*

*"textSize" : "19"*

*},*

*{*

*"noteType" : "Free",*

*"noteFont" : "Verdana",*

*"textColor" : "Black",*

*"textSize" : "19"*

*},*

*{*

*"noteType" : "Free",*

*"noteFont" : "Verdana",*

*"textColor" : "Black",*

*"textSize" : "19"*

*},*

*],*

*}*

/api/notes/<ownerid>/favs - Fetches fav notes for a specific owner

Example:

*{*

*"133453" : [*

*{*

*"noteType" : "Free",*

*"noteFont" : "Verdana",*

*"textColor" : "Black",*

*"textSize" : "19"*

*}*

*],*

*}*