Low Level Design (LLD)

Notes Keeping APP

Written BY	Ngamlenmang Touthang
Document Version	1.0.8
Last Revised Date	02/02/23

Document Version Control

Change Record:

Version	Date	Author	Description
1.0.0	11/01/23	Ngamlenmang	Server initial setup
1.0.1	12/01/23	touthang	User authentication added
1.0.2	13/01/23	Ngamlenmang	Notes controller and routes added
1.0.3	14/01/23	touthang	Swagger docs added
1.0.4	15/01/23	Ngamlenmang	Client template with responsive added
1.0.5	16/01/23	touthang	Client side authentication UI added
1.0.6	18/01/23	Ngamlenmang	Search feature added
1.0.7	19/01/23	touthang	Cookie update
1.0.8	02/02/23	Ngamlenmang	Documents updated

Reviews:

Version	date	Reviewer	comments

Approval Status:

version	Review Date	Reviewed By	Approved By	Comments

Contents

1. Introduction

1.1. Why this Low-Level Design Document?

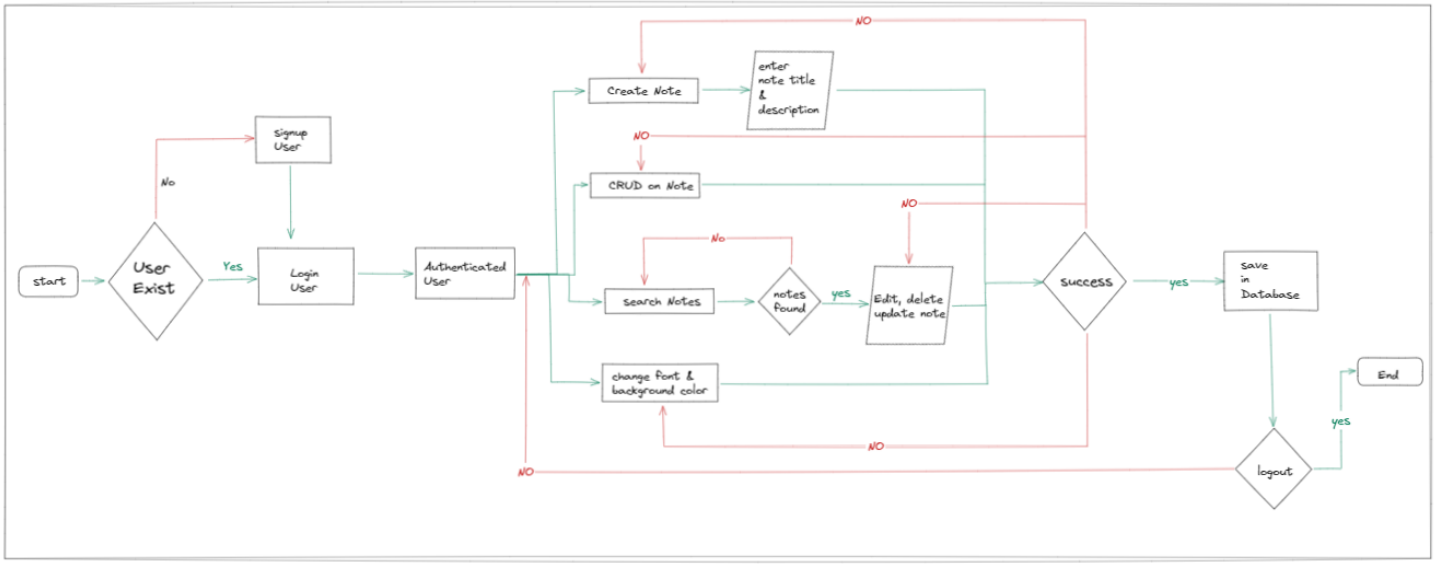
The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model of coding. The document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

1.2. Scope

1. Everyone overlooks crucial items in this hectic world, but adopting a note-taking software can help. Many users will benefit from this app by having a complete list of their chores and stuff in one location.

2. The user will feel at ease interacting and using it because to the straightforward but interesting user interface

2. Architecture



3. Architecture Description

3.1. PlatForm

- The keeper notes application web based application and different responsive screen size.
- It will be cross platform.

3.2. Data storage

- MongoDB Atlas will be used as database management system to store notes and user information.
- The data store in database will be store as Document (NoSql) and mongoose for making the design schema

3.3. Security

- Bcryptjs and Jsonwebtoken is used for encryption and secure the storage of sensitive information (e.g. user passwords)
- User authentication and authorization mechanisms

3.4. User Interface

- User interface design and layout with Reactjs library
- User interaction design

3.5. Network

- Data transfer mechanism is mainly done via REST API

3.6. Performance

- Scalability and performance optimization techniques
- Load testing and stress testing procedures

3.7. Technical Dependencies

- Some of the third party libraries, frameworks and tools to be used in development are – bcryptjs, cookie-parser, cors, dotenv, express, jsonwebtoken, mongoose, morgan, nodemon, swagger-ui-express, validator, yamljs, reactjs, react-toastify, react-color and axios

3.8. Logging

- The react-toastify log ever success and failure events that occurs in the application

3.9. Deployment

The Notes keeper app is deployed using Railway (backend), Netlify (frontend) and database (MongodDB Atlas) via github which is software version control system

4. Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
Verify whether the application URL is accessible to the user	1.Application URL should be defined	Application URL should be Accessible to the User
Verify whether the application loads completely for the user when the URL is accessed.	1.Applicaiton URL is accessible 2.Application is accessible	The application should load completely for the user when the URL is accessed.
Verify whether user is able to Perform signup and login	1.application is accessible 2.User signup and login	User should be able to enter city name in the input field
Verify whether user is able to create note	1.Application is accessible 2. user is authenticated 3. note create form should be there	User should able create note
Verify whether user can retrieve a single note	1. Application is accessible 2. User is authenticated	User should be able to fetch single note data
Verify whether user is able update a note	1. application is accessible 2. user authenticated	User should be able to update a note
Verify whether user is able to delete note	1. application is accessible 2. user is authenticated	User should be able to delete note.
Verify whether user is able to search for specific note by title	1. application is accessible 2. user is authenticated	User should able to search a particular notes by title

Verify for proper handling of unauthorized access of notes	1. application is accessible	Notes should only be access by the individual (owner) .
Verify for handling of invalid data inputs while creating or updating a note	1. application is accessible 2. user is authenticated	All invalid data input must be handle.
Verify for error handling when an unexpected error occurs	1. application is accessible 2. user can signin/logout	Any unexpected error occurs should be properly handled.