Project Report - ShareNote

Team Worf: Khelan Natha (200352840), Alok Paranjape (200246370), Vandan Chauhan (200465976)

Background:

Taking notes is an easy task. A person can take notes on a notebook, a page, or in their computers, tablets and any devices of their choice. But being able to share the notes they have taken is a challenge.

To overcome this, services such as messengers, Discord, etc are often used to share the notes they have taken. So wouldn't it be good if such a service was developed? A service that could enable the students to make notes and share them automatically.

Project Idea:

To create an app that allows users to effortlessly share notes with relevant peers. At the same time, it offers users the ability to quickly and efficiently take notes, while keeping everything organized and tidy. Notes aren't strictly the longform pieces that are written alongside meetings, like one might do with Notability. And lastly, users can chat with each other, with the option to flag just the important 'tid-bits' for everyone else (these flagged chats in our opinion are also considered as 'notes').

Goals of the Project:

The ultimate goal of this project is to make taking and sharing of notes easy for the users and at the same time, combining both the functionalities in one single website. This will, in the end, help students get rid of the tedious efforts that had to be made to share their notes. Additionally, it'd be a benefit for the students if they are able to view the flagged notes in chat, or their respective class group.

The ultimate goal of this class is to learn about the various stages of a project life cycle and be able to produce some minimum product that is useful enough for our target audience to provide feedback.

Goals of the project that were achieved:

The functionality of the users being able to take and share notes on the website were successfully achieved. This was one of the basic things needed on the website. The users can take a note, and save the note in a folder. This would enable sharing it with all the users that access the particular folder where the note is stored.

This is a part of what was envisioned in our grand scope. We weren't to implement the full user experience for sharing functionality. i.e where the user could decide either to share or not to share the notes. Also for the time being, the user isn't able to share the taken notes with a particular user. Additionally, we weren't able to implement the flagging notes in the chat feature, particularly because of the time and skill constraints.

Project Initiating:

First of all, we developed the conceptualization of the project's idea. To do this we used the principle of 'The Golden Circle'. This is a concept put forth by Simon Sink that argues the leading companies in the world all have great products because they first address 'Why' they do something, then 'How' they do it, and finally addressing the 'What' referring to the product itself.

The Golden Circle made us answer the basic questions, such as;

Why? - We think learning should be an efficient and collaborative experience, instead of an isolated personal obligation.

How? - We aimed to do this by making it easier for students to share notes. The final outcome would also enable the users to see flagged notes in chat boxes.

What? - Our project would be a note taking web app that can be used to share notes with peers, and access the shared notes of other users.

Additionally we addressed the question of 'Who?' to address our target user base - The target audiences for this app were primarily students, since taking notes is important to them. Educators and tutors are also a target group for this project since they can share notes with students.

Project Implementation Approach:

It was defined that the tasks had to be done one at a time, such as planning(documentation), software design and architecture (diagramming), software construction (execution), and finally ending with delivering a Minimum Viable Product. Thus a process that related to chunks of work rather than the entire work at once was considered as the best for this. Keeping this, the requirements and constraints of the project in mind, Incremental & Iterative approach was decided as to be the best suited.

Chunks of the work for this project were time driven. And as they kept on getting finished, they were integrated into the system, which later transformed into a working project.

It was decided to respond to a change or a request of change whenever required, and all of the members wanted to work together on all the deliverables of the project. Additionally, a Minimum Viable Product (MVP) was needed to be constructed. Hence we decided to add an Agile approach as a part of our project. *[However, one thing to be noted is that we didn't create working software at all the stages of the project, which happens to be an Agile principle]*

Project Planning - Documentation:

For the planning phase, we explored several documents such as;

Stakeholder Engagement Plan & Analysis Document:

Stakeholders are the people or entities who can have an effect on or be affected by the project or its development methods. A plan had to be devised for having a proper way to handle all the stakeholders. Firstly, the stakeholders were identified for the project. Students were the primary stakeholders, since they were the primary audience of the app. Also, educators and tutors were added as stakeholders. Later on in the project, our Project Manager and Sponsor, Tim, was also added to the stakeholders list. Following this, a plan was devised to engage with our stakeholders as per the needs of the project. They were categorized in three sections, i.e. Pay Close Attention, Meet

Requirements and Monitor. The project manager needed to be met with requirements, since the work had to be handed to him. The students had to be paid close attention to since they were the primary target audience. The Educators were monitored since their opinions might have had an impact on the project.

Business Case Document:

First thing done was to identify the need of the project, i.e. the business need. For this, the Business Case Document was drafted. Notes can be easily taken on online platforms, personal devices. But that issue arises when they are to be shared. The primary goal of this app was to allow the users to share the notes they took. Also looked upon were the possibilities of either creating a Website based app or an app that would be usable on Desktop/Mobile devices. However it was concluded that would be best suitable to build a Web based app, because of the time constraints. Also, the scope of a Desktop/Mobile device based app would have been huge, and well far out of reach for us regarding the skill constraints.

Project Charter Document:

It was decided initially that the deliverables defined in the Scope Document should be attained at specified dates. This proved to be rather undoable since the class content and the project proceedings didn't align together. Additionally, several risks were identified that could be encountered during the project planning and execution phase. All of these things were noted in the document alongside a rough budget that just accounted for the manual hours pricing for the project.

Roles and Responsibilities Document:

Since it was a group project, each member would have a certain responsibility. As per the concepts learnt in the class, either one of the two routes could have been chosen. One where everyone is a generalist and everyone contributes, in one way or another to every aspect of the project. Another is where everyone is given a specific role, and their contribution is limited to that particular thing only. For this project, the generalist roles were chosen, since everyone wanted to be a contributor.

Project Scope Statement Document:

A Project Scope Statement was created in order to identify and document various deliverables of the project. All the high-level deliverables were noted. Also noted were the things that were not being included in the scope for the project, but would be added if time and skills permit.

RACI Chart Document:

Since every member wanted to contribute, it was only good if all the members were familiar with all the aspects of the ongoing things in the project at a given time. Hence, it was decided that every member should be consulted and informed of the ongoing activities in the project. Since this was an approach where everyone was contributing, it was better that everyone was responsible for the things as well. And thus, the members can also take accountability for those things. Hence, everyone was considered to be accountable and responsible for the project activities.

User Stories:

User stories are written to get a general explanation of a software feature from their perspective. This helps understand how a piece of software[work] would be important or would impact the user. It was decided from the User's perspective to create such stories that could help better understand the requirements for such a project.

Creating an Minimum Viable Product(MVP):

A Minimum Viable Product (MVP) is a version of the product that has just enough features for the user to use and to give feedback. A MVP excludes unnecessary features. Thus, a MVP was decided, which included: being able organize Notes, see important flagged chats (from the chat box) as notes, share notes with peers, save and access the notes online, create an account for the notes.

As per the discussion, we decided that those were the minimum requirements for an MVP.

Software Design & Architecture - Diagramming:

Class Diagram:

A class diagram encompasses the structure of a system by showing the system's classes, their attributes, the operations, and the relationships among the objects. For the Class Diagram, we had thought of it as of much value, but when it came to designing, we found that it wasn't much of a use for our project in particular. We had an understanding of our project and the Class DIagram didn't add much to it. The time spent on that could have been used to design a better MVP.

Data Flow Diagram:

Data Flow Diagrams is a Structure outlining various paths of data flow that would be involved in the project. This can be done by identifying all the inputs and outputs in the system, and connecting them with the data that flows between them. It was pretty much helpful in the understanding phase. However, during the execution phase, a significant chunk of it didn't line up with the implementation. But overall it was quite useful from the understanding perspective.

Mode, View, Controller Diagram:

A Mode, View, Controller Diagram is a diagram where the concerns of the system are separated from the programmer's perspective. The Model deals with the data, i.e. handles it. The View component represents the data and the Controller enables interconnection between View and the Model. From the perspective of a project, this would have been more useful if all of the members were working on separate aspects of the MVC. However as it turned out later, there were limitations to who can work on what. Hence in the end, it didn't have much value to us with respect to the implementation.

Use Case Diagrams:

A Use Case Diagram is designed to summarize the Users and their interactions with various components of the system. For our project, we thought that it would be a good way to envision the overall site structure and implementations. However, the actual implementation proved to be very different. This probably would have been as a result of our inexperience.

Lo-Fi Sketches:

The importance of Low Fidelity Diagrams was understood from the very moment it was taught. These are diagrams that show how the User Interface of a website would look. For this project, the sketches were designed in a way that encompassed the ideas of all the members. Also, it was done with little effort.

Updates To MVP:

After doing the diagrams, we refactored our MVP slightly, removing organizing notes, as that falls outside of the minimum functionality necessary for making and sharing notes. Also it was decided to

let go of the login and Signup functionality. But upon discussion it was found out that it has importance in the MVP.

Software Construction - Coding:

Just as this phase of the project was encountered, it was understood that all the members of the team wouldn't be able to work with the execution. This was particularly due to lack of skills. Also there were several issues regarding the working of code.

Initially, the team was naive to how much coding work it would actually take to build out even just a barebones Minimum Viable Product.

- So Alok began working on a Node.js implementation of the ShareNote app, while I (Khelan) worked on the PHP implementation.
- Through our inexperienced idealistic lenses, at the time this seemed like a good way to explore which method results in overall better user functionality and actual implementation.
- This decision ended up working against us in the end. We learned that splitting our efforts across two different implementations took away from how much we would have been able to get done on just one collaborative implementation.

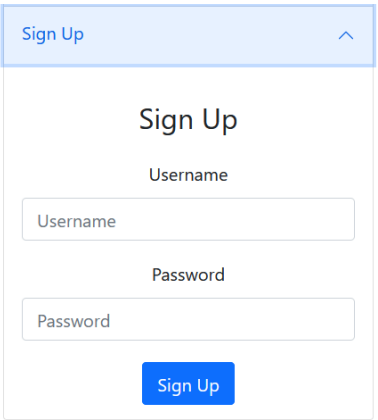
Rationalizing the MVP during the execution:

When the execution phase was going on, we understood that there were several constraints. Time and skills being the foremost. There were only a few weeks left for the final deadline, and hence it was decided to let go of some features.

This included the chat flagging functionality, as we thought it wouldn't be achievable in the available time. Alongside this, the sharing functionality also had several issues with the execution. Hence a modified version of it was included in the MVP.

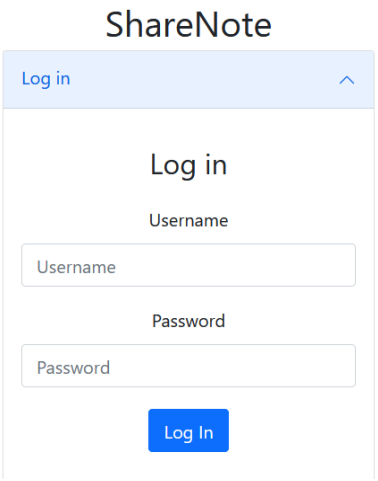
Web Application Screenshots:

Log In & Sign Up:



Login and Signup are features of the Final and rationalized MVP that enabled the user to create an account and Log In to an existing account.

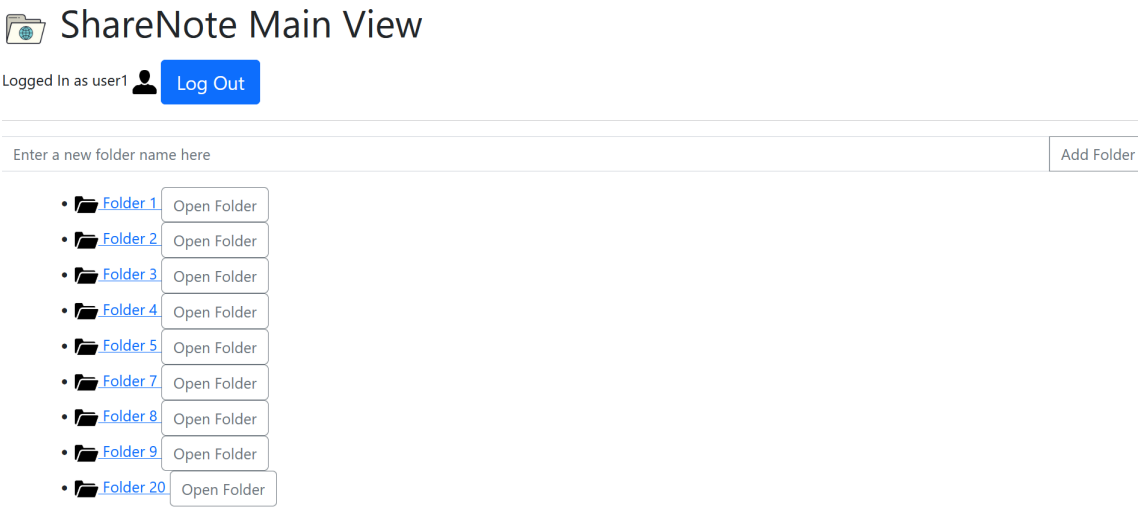
Existing Users can Log In and New Users can Sign Up



Folder View:

Being able to view the Folders where notes are stored is an important aspect of the MVP since it is necessary for accessing the notes. All the notes are stored in Folders. The users can navigate through different folders to access the notes that were created by them and other users.

This is the page where the User lands upon Logging in using their credentials. This page views various folders that have been created by the user, as well as other users. This page also allows the User to create a new folder by entering the desired folder name in the text bar and pressing the Add Folder button.

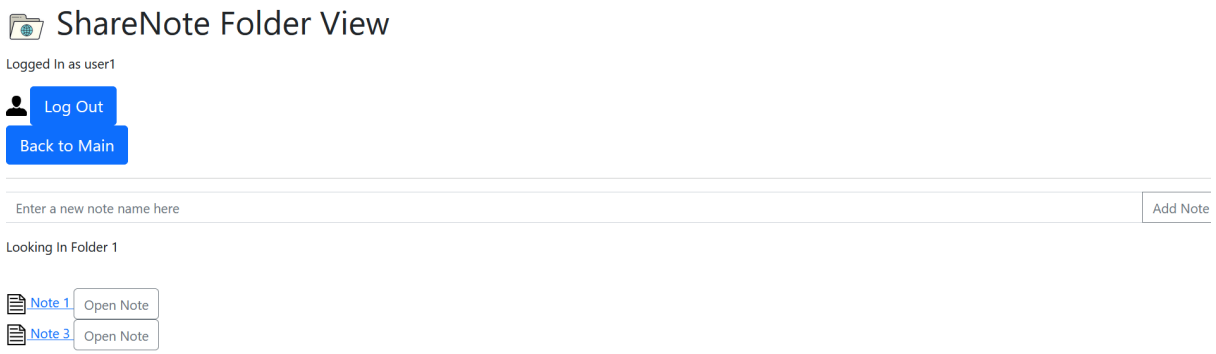


Folder View / Note Editor:

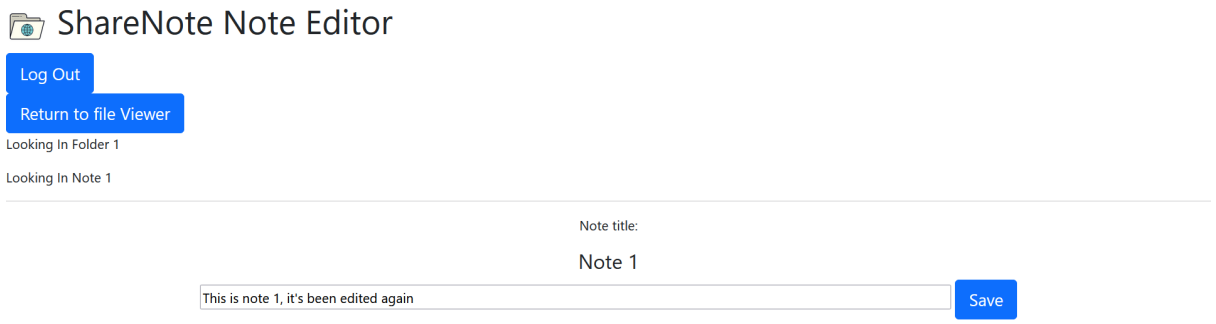
This is the part of our MVP where the user can take notes. The user would also be able to view and access other notes that are present in the folder or created.

This page is seen when the user clicks on a particular folder of their choice. The view contains all the notes that are present in the folder, if there are any. If the folder was just created by the user, it won't have any files in it.

Also in this view, is the text area where the user can write a note.

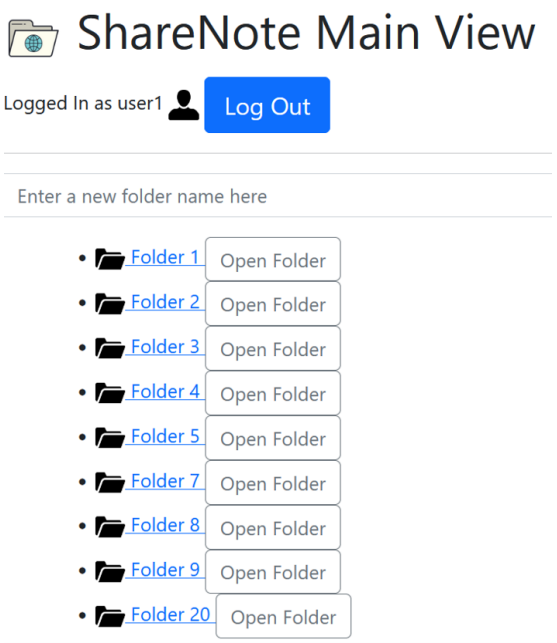


Upon clicking the text area, the user can write a note. The note written can be saved by pressing the Add Note button.



How is the Sharing functionality implemented?

The User can then click the Return to File Viewer button. This would land the user back to the Folder View page. When a different user logs in using credentials, they will be able to view the note created by another user, by navigating through the folders.



Project Sponsor (Instructor) & Peer Feedback:

Questions regarding the unclear Roles and Responsibilities:

We had decided to follow the Agile approach, where everyone on the team contributes equally to all aspects of the project. However it turned out later that for our team, there were quite a few constraints regarding who is able to do what. So we decided to divide the roles and let people do what they can do the best, while the others try lending a hand alongside doing their own bit. Therefore, by the end everyone had clear Roles and Responsibilities.

Possibility of the website getting converted into a platform that is frequently used for cheating:

There have been several websites that are used these days just to take notes. But after the feedback received from our Project Manager Tim, we decided to look up on those. We saw such websites that were directly or indirectly used as sources to get answers. Some websites had emerged as a note taking platform, and the notes were actively and repeatedly used for plagiarism. However we concluded that if implemented, this was too big for our scope.

Improving Github readability such that it is understandable to people outside of the class:

The feedback was taken into consideration and the names of the directories were changed to more meaningful ones.

Rigid Milestones (peer):

The milestones decided by us were just us thinking a bit and coming up with the dates.

Details not needed in vlogs since content is already seen (peer):

Taken into consideration, the succeeding vlog had less discussion on the slide content and more of what it meant to us.

Team Reflections:

Include a deeper reflection and lessons learned as teams discussed in the final demo/presentation (described below under "Presentation criteria"), including a deeper reflection on all stated questions and other things like how the team/individuals overcame obstacles and frustrations (be humble, open, honest).

How did you feel about this project? Do you think your team was successful, or? What did you like about the project? What did you dislike about the project?

- We felt this project was quite helpful for learning about project planning and implementation. We got to learn about team coordination. As a team we think we were moderately successful overall. We enjoyed working as a team. One aspect we weren't too fond of was how we are expected to make judgement calls about how best to handle the project, but without experience you end up making costly mistakes leaving you in a time crunch. But we chalked this up to being part of the learning experience.
- In all honesty our final product could have been much stronger if we had made different calls on a few critical decisions along the process. This is not to say that we don't feel we were successful, but with hindsight being 20/20 it's easy for us to now look back and see how we could have done things better. More specifically, we wouldn't have wasted so much time on the software design phase. The class diagrams, use case diagrams and

data flow diagrams were extremely complex and time consuming. They also ended up not being followed because when you start coding it's extremely difficult to follow these ideal diagrams prescriptively. For our level of project and coding experience (completely green beginners), I would say the two most important aspects that we should have devoted the most time to were: 1. first defining, and then continually refining a true MVP with as limited a scope as possible, and 2. diving into the actual coding, or software construction phase of the project.

What did your team feel most proud of throughout the entire project experience?

- Despite our inexperience and the difficulties we surmounted, what we did was a huge achievement.

What did you learn about yourself as you collaborated and worked on this project (individually and as a team)?

- Khelan - I learned that it will take practice and experience to get good at doing projects. The biggest takeaway for me in all of this was the cumulative learning experience. I say this because when you first start out each project activity, you have this ideal best-case view of how things will pan. This coupled with low experience in actually coding/implementing the project solution ends up leading to overly optimistic goals and a lack of time. I think it would be more helpful if the software construction part happens sooner. This is the critical phase of the project where you really start having your eyes opened to what is really needed in the MVP, and how much time and effort these things really take. If we go through this rude awakening sooner, then it might leave us more time to be able to adapt and still produce a strong product by the end of the class.
- Alok - I learned sometimes I freeze up when I deal with intimidating tasks, but that it's better to try something, than to give up. There were a lot of difficulties when coding this project. I could have given up and said that the task was too difficult to complete. What I really internalized when developing this project, was to break the initially impossible task, down into smaller pieces. Instead of looking at the project all at once, I started by emulating what happened in the labs and incrementally building it up from there, going from just uploading text to having it fully connected to the database. As I keep doing the program, I know I'm going to face similar difficulties, so I should keep applying what I learned.
- Vandan - I learned that there may be issues, but always do your bit and contribute as much as we can. Looking back, I can say that I didn't have much experience with software project planning, coding. Also, shifting from a significantly different field of study to Computer Studies didn't turn out to be as yielding as it could have been. But a start is a start, this is the way I learnt. I felt inadequate, not being able to contribute more to the execution phase.
- As a group we learned that it's better for people to specialize rather than everyone being a generalist - but we also learned that this was hard to assess in the beginning because of our inexperience

How will you use (or not use) what you have learned/experienced in this project going forward?

- We definitely will use the knowledge we gained related to planning when we go out in the industry, but it'd be project specific. We'll probably need it for other classes, or when making our own websites.
- Won't Use
    - Class, data flow and use case diagrams, at least not until having undergone some basic implementation and/or having more experience.
- Will use
    - The experience with web development as a whole and project management techniques.

What "stuff & things" related to this project would you like more help with?

- We'd like more help with managing expectations for our MVP, and feedback on if a proposed MVP is attainable for our coding experience.
- More documentation and examples with either of the web development implementations used to develop the product, as well as GitHub
- More feedback in regards to how the documentation we made would tie into the later sections of the project?