Team 2 Time Series Repository
Project Plan

Madeline Porcaro, Luying Cai, Catherine Raj, Sam Schwartz
Team 2
https://github.com/orcap23/CS422Project.github.io
J. Flores - 10-12-2021

Time Series Application

With the rise of machine learning and data science, along with the places one would find machine learning and data science, even in places one wouldn't expect. Data drives society as we know it today. It gives us recommendations on driving directions, where to eat, what to buy, who could be a romantic partner. With this rise in information about the world around us or about us, the data being collected needs a home. Team 2's Time Series Repository (T2TSR) is that home. T2TSR is a web-based application that holds training datasets along with the functionality of uploading datasets to see statistical comparisons between a base set and the uploaded set.

Management Plan

Our team was divided into three teams, server side, client side, and documentation. We divided the team up this way so there was at least one person who specialized in each major component of the project and their knowledge could be communicated across the team. Our architecture is as as follows:

- Client side (what the user sees)
    - Homepage
        - Allows users to access the repository and download from the repository.
        - Download button
    - The Datasets
        - These are downloadable datasets for users to interact with along with the statistics of each dataset that is held in the repository.
        - The user should also be able to upload their a set to compare to a dataset on the repository
        - The statistics should change after an upload is made.
- Server side (what the programmer sees)
    - Database
        - User Uploads
            - Public
                - Name
        - Downloadable Datasets

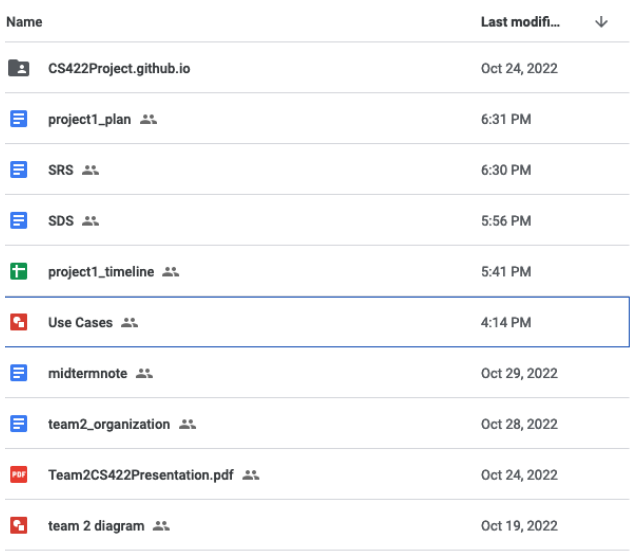
        - TS data
        - TS metadata
- Uploaded Datasets/ Model Data
    - TS data
    - TS metadata

Instead of a team of five, we were equipped with a team of four. On the client side, Luying built the HTML skeleton, event listeners (buttons) along with connecting the back-end to the front-end and any requests to change pages (i.e. from Home-View to Modeling-View). Catherine was assigned the UI/UX along with some of the tasks Luying took on as she was consistently unavailable and not meeting deadlines.

On the server side, Luying was assigned building the databases for the uploads and downloads, getting the datasets that were to be downloaded, and building the framework for the repository. Sam was assigned the modeling forecasting, statistical analysis of files being read into the system and file verification.

On documentation, that was assigned to Maddie. She made the software designs, use cases, SDS, project plan (this doc), the presentation, the timeline and aside from documentation, she handled the organizational structure by creating the shared Google Drive, Git repository and the main Discord group for team members to communicate. The main Discord group was for the project lead to get updates from the team and lay out expectations for the team over the course of the weekend or throughout the week.

The shared Google Drive looked like this, as seen on the right.

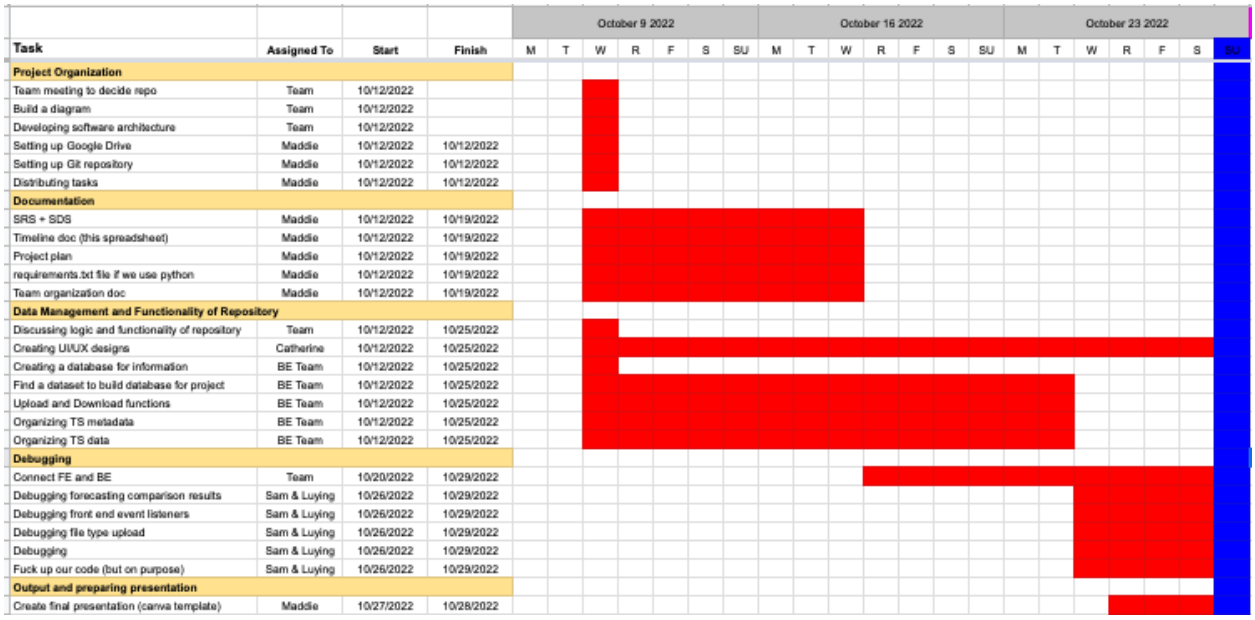


It was common for Luying and Maddie to assist the other in their tasks.

It was pretty difficult to get the team to meet all at once due to members not communicating their availability and team members not responding to messages in a timely manner. A large amount of communication was done between Luying and Maddie to get progress reports to keep the documentation and the code consistent.

Project Timeline

In order to deliver a product that was both functional and of quality, the team built a timeline and divided tasks among themselves. Our project timeline is shown below:



Task	Assigned To	Start	Finish
Project Organization			
Team meeting to decide repo	Team	10/12/2022	
Build a diagram	Team	10/12/2022	
Developing software architecture	Team	10/12/2022	
Setting up Google Drive	Maddie	10/12/2022	10/12/2022
Setting up Git repository	Maddie	10/12/2022	10/12/2022
Distributing tasks	Maddie	10/12/2022	10/12/2022
Documentation			
SRS + SDS	Maddie	10/12/2022	10/19/2022
Timeline doc (this spreadsheet)	Maddie	10/12/2022	10/19/2022
Project plan	Maddie	10/12/2022	10/19/2022
requirements.txt file if we use python	Maddie	10/12/2022	10/19/2022
Team organization doc	Maddie	10/12/2022	10/19/2022
Data Management and Functionality of Repository			
Discussing logic and functionality of repository	Team	10/12/2022	10/25/2022
Creating UI/UX designs	Catherine	10/12/2022	10/25/2022
Creating a database for information	BE Team	10/12/2022	10/25/2022
Find a dataset to build database for project	BE Team	10/12/2022	10/25/2022
Upload and Download functions	BE Team	10/12/2022	10/25/2022
Organizing TS metadata	BE Team	10/12/2022	10/25/2022
Organizing TS data	BE Team	10/12/2022	10/25/2022
Debugging			
Connect FE and BE	Team	10/20/2022	10/29/2022
Debugging forecasting comparison results	Sam & Luying	10/26/2022	10/29/2022
Debugging front end event listeners	Sam & Luying	10/26/2022	10/29/2022
Debugging file type upload	Sam & Luying	10/26/2022	10/29/2022
Debugging	Sam & Luying	10/26/2022	10/29/2022
Fuck up our code (but on purpose)	Sam & Luying	10/26/2022	10/29/2022
Output and preparing presentation			
Create final presentation (canva template)	Maddie	10/27/2022	10/28/2022

Building Plan

Team members were assigned tasks in a more broad sense and it was on the engineer to break down that task as they saw necessary. The first stage of development involved meeting over Discord to discuss the layout of the project and assign tasks to each member. We talked about what we needed to know on the technical side or development, this included finding resources that we needed to know to build our system. The second stage was spent building the framework for the system along with making a more in-depth software design to help fill out the framework. We met with our client to give them updates on the progress and the changes in team dynamics as time went on. We also spoke with our client to get further clarification on expectations or requirements that had been mentioned and getting more detail on the requirement. By this point in the project we had a solid foundation of the technical pieces needed to build the project, it was a matter of learning the required information to get the project built. By the final stage of the project, the team was adding any last minute details such as finalizing the UI/UX, making sure the documentation and code reflected each other well and any last changes to the model testing in the backend. The team was also debugging throughout the process to make sure that any problems arose were taken care of swiftly and to ensure no snowballing of issues would appear towards the final deadline.

Monitoring and Reporting

Our approach to monitoring was the project lead would send a message twice a week based on the development they saw in the repository. They would lay out what was expected from the front-end and the back-end over the course of a few days and would ask for updates from each team member on an almost daily basis. The timeline was made and the team was expected to stick to it, although it was often seen that some team members would not refer to the timeline and more often than not say anything about missing deadlines or lack of progress until prompted by another team member. Based on the lack of information, deadlines were changed and that was reflected in the timeline. This posed a great challenge in development since some team members would finish their section of development and see that their module could not be tested to be compatible with other modules as they would be incomplete.

Rationale

We decided to break down our system and establish as much clarity among the team on the expectations for each role and what was expected from each team member. One of the biggest challenge was the development gap between the front-end and the back-end as the main back-end engineer needed to have some semblance of a front-end to test their functionality, like uploading and downloading so in turn, they built a large part of the front-end skeleton so they could test their work. We needed to collaborate and communicate often on the work that was being done in the different areas of development.