Team 2 Time Series Repository
Software Design Specification

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

Table of Contents

1. SDS Revision History

Date	Author	Description
10-12-2022	Maddie P.	Created the SDS documents
10-14-2022	Maddie P.	Added system overview
10-15-2022	Maddie P.	Further edited system overview
10-17-2022	Maddie P.	Added software architecture
10-21-2022	Maddie P.	Edited software architecture and software modules
10-30-2022	Maddie P.	Finalized SDS and added revision history

1. System Overview

With the rise of machine learning and data science, it's often difficult to navigate the sea of data that is accumulated through research, data mining and other means of data collection. Anyone can go through data and various datasets but going across multiple platforms can be tiring and overwhelming. That is the intention of Team 2's Time Series repository, to hold datasets with the characteristics of a time series. The intention of this repository is to alleviate the hassle of finding training sets for time series forecasting.

Team 2's Time Series Repository consists of three major components: The application, the database and the user interface (UI). The application will be able to have users download the dataset they desire and use it for training their model as they see fit. There is also a functionality for uploading to see the changes from the original dataset that they download to the new set that had been generated through adding noise and training what model they needed to train. The database holds the training sets to be downloaded. The software that creates the database is not seen by the user. The UI is the only item seen by the user. The UI is the platform from which a user can download the training sets and upload their changed sets.

2. Software Architecture

2.1 Components and their functionality

- 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

2.2 Module Interaction

Within the repository there are a number of modules. They are all equally crucial to making the software work. There are two modules, Home-View (Homepage), Modeling-View (Uploads).

Home-View is the main page after connecting to the web-page. This is the main area for users to see the datasets that can be downloaded. Users have the ability to download datasets from the homepage and use them to their needs as training sets. When the user wants to download from the repository, the files that can be downloaded are in the form of a zip file, containing the training set, the time series itself and forecasting data.

Modeling-View is largely an extension of Home-View but it holds the capability to handle the uploads from the users. It reads the files that are uploaded. Modeling-View reads the file and compares it to the test stored in the Django database. Modeling-View also shows the model comparison and testing, they are updated as the upload to the Django database happens. It also has the ability to let data scientists or machine learning engineers add their name to the datasets they upload and also delete datasets that are uploaded.

2.3 Rationale for Architecture

Team 2’s current goal is to allow access to datasets for everyone so training machine learning models can become more streamlined and accessible to more individuals who can try their hand with machine learning and data science with the skyrocketing interest and various applications that have appeared in the last decade or so. There are two major modules, Home-View and Modeling-View.

Separating Home-View to give some information about the repository and the downloads for each dataset in the database and Modeling-View for uploads and changes for the modeling views after uploading gives the separation so a user is not overwhelmed with the amount of information on a given webpage and the separation of the downloadable datasets and uploaded datasets in the database was to make the jobs of the database administration and backend engineers easier. Separation is a big theme across the system so the system is broken down into small, manageable parts for both the users and the engineers.

3. Software Modules

3.1 Home-View

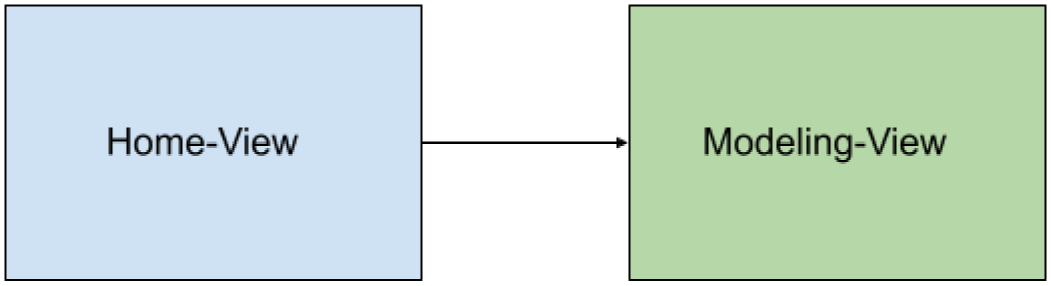
3.1.1 Primary Role and Function

The Home-View is the main landing page for users. The user will always start at the Home-View. The Home-View holds the information about the repository and it holds the download buttons that access the Django database for the training sets that can be downloaded. For demonstration purposes, there are only two datasets that can be downloaded but there is a framework in the backend that can be expanded upon to add more datasets to be downloaded and used as a training set.

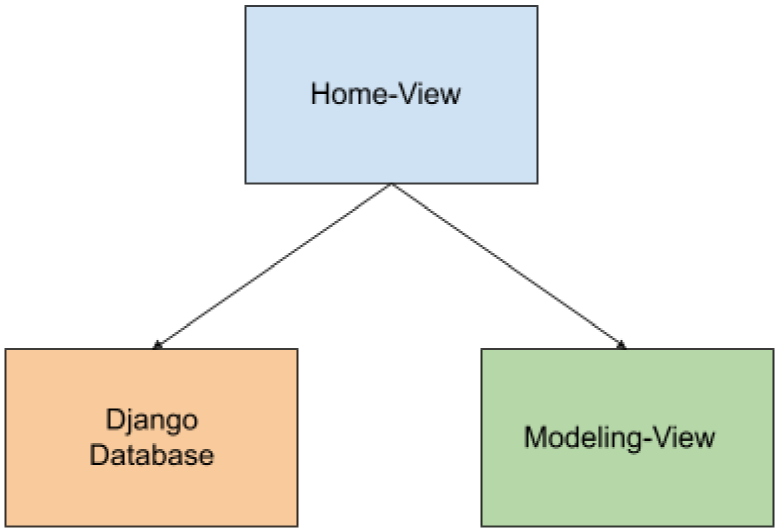
3.1.2 Interface

The interface for the Home-View is simple. There is a table that has the downloadable datasets that can be downloaded with information about each dataset, including name and and some information about each dataset. It is open to all users, free to download. There is a link to Modeling-View so users can upload their datasets for comparison and see statistical differences between the base set and their uploaded set.

3.1.3 Static Model



3.1.4 Dynamic Model



3.1.5 Design Rationale

To ease the users, there is a separation between the Home-View and the Modeling-View. As a general rule of thumb, there are as few clicks to navigate the website. Models were based on the Monash Forecasting Repository which keeps its message about the repository and their work and its download page different, but since Modeling-View has an upload, the download and upload were separated to make sure users knew what they were clicking on and knew where they were and the functionality of Home-View.

3.1.6 Alternative Designs

Another possible design for Home-View would have been to have a Login service if users wanted to keep track of their downloads and “star” or “favorite” any repositories that may have been of interest to them but they maybe didn’t want to download.

3.2 Modeling-View

3.2.1 Primary Role and Function

The main idea behind Modeling-View is there is an upload function for users to upload their own datasets to be compared with a base dataset that the users cannot see or access and there are visuals that will show them items that can be statistically measured including mean and standard error. This page is linked to the Home-View or the main page and can be accessed from the homepage. Modeling-View holds the information about the metadata for datasets along with names of users who have uploaded.

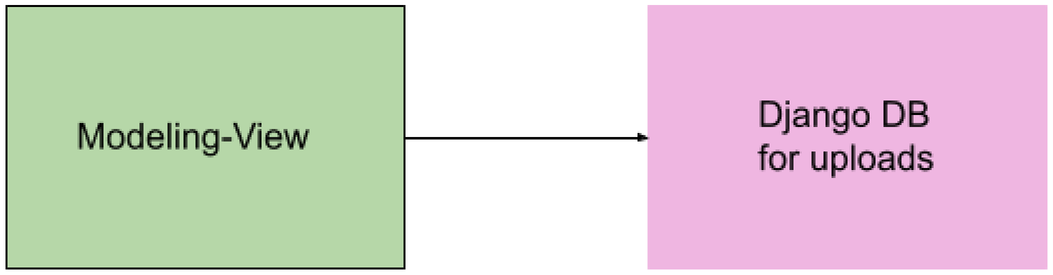
3.2.2 Interface

There are few user interactions in the interface, a functionality for uploading. Uploads from other users can be seen, but are not accessible since they are not equipped with the download capability. There are some key pieces of information in Modeling-View:

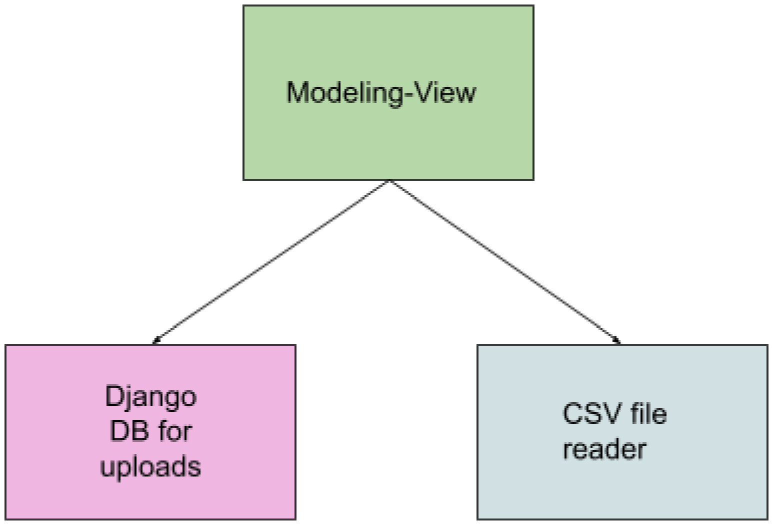
- ☐ Users' uploads
- ☐ Evaluated statistic values
- ☐ Metadata for datasets
- ☐ Delete functionality

Users have the option to remove datasets that they have uploaded along with uploading and see the evaluated statistic values from the uploaded dataset. Modeling-View only takes CSV files so any files that are not a CSV are rejected.

3.2.3 Static Model



3.2.4 Dynamic Model



3.2.5 Design Rationale

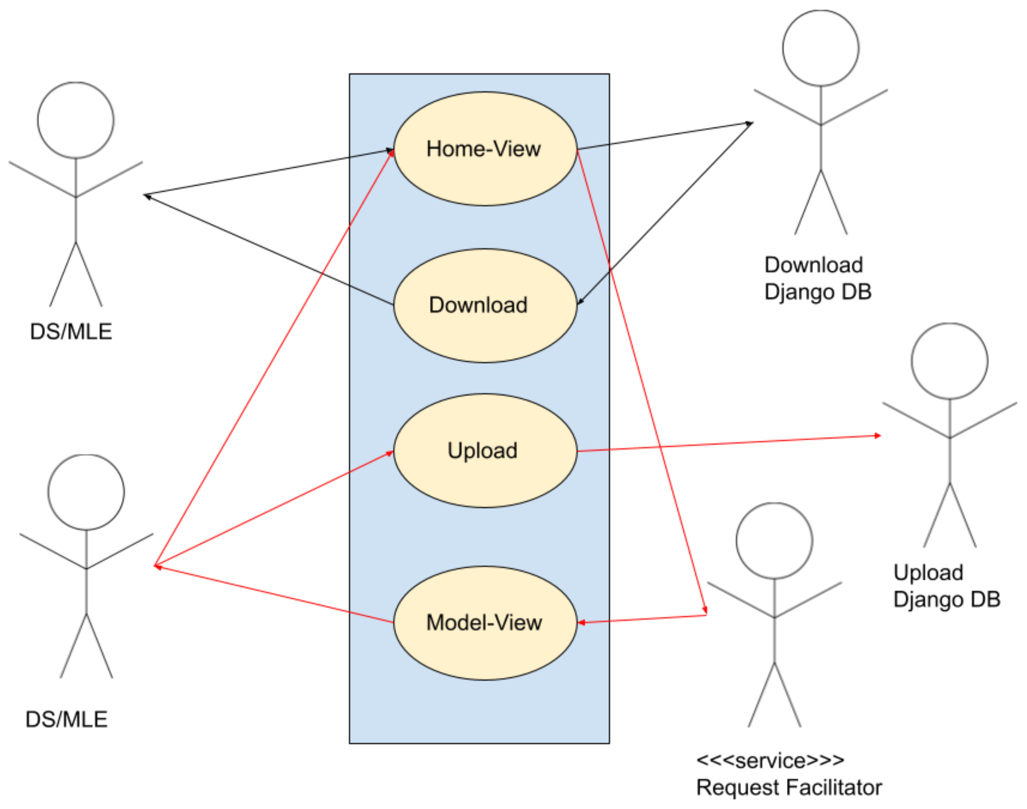
To keep users from getting confused about uploading and downloading, there is a separation between Home-View and Modeling-View. There is also a separate database for the user uploads for database administrators and backend engineers to be looking at the correct datasets for their own work.

3.2.6 Alternative Designs

Another possible approach to the Modeling-View is there could be more robust visuals in evaluations for the users to see the differences in graphs and charts rather than reading numbers on a screen. The delete button could also only show for users' own datasets and not all datasets that have been uploaded by other users.

4. Dynamic Models of Operational Scenarios (Use Cases)

There are two major uses for this system. One is outlined with black arrows and one is outlined with red arrows.



5. References

Faulk, Stuart. (2011-2017). CIS 422 Document Template. Downloaded from https://uocis.assembla.com/spaces/cis-f17-template/wiki in 2018. It appears as if some of the material in this document was written by Michal Young.

IEEE Std 1016-2009. (2009). IEEE Standard for Information Technology—Systems Design—Software Design Descriptions. https://ieeexplore.ieee.org/document/5167255

Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. *Commun. ACM, 15*(12), 1053-1058.

Class Diagram. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/Class_diagram.

Sequence Diagram. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/Sequence_diagram.

6. Acknowledgements

We would like to acknowledge Juan Flores for his support throughout this project and Kartikeya Sharma for sitting in on meetings during the software development process.