TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Interim presentation master thesis

Topic

Comparison of data-based signal generation using machine learning and time series decomposition.

Felix Fischer // 26. Oktober 2023

Task

1. Research and analysis of existing literature and current possibilities to generate data based on existing datasets, with a special focus on:
   - Time Series Decomposition, and
   - Machine Learning Algorithms
2. Identification of requirements for software that simulates data based on an existing dataset.
3. Conceptualization of a solution to generate data using Machine Learning to simulate sensors.
4. Extension of the existing prototype for sensor simulation with a data-based approach to create the calculation rule for sending data based on:
   - Time Series Decomposition, and
   - Machine Learning.
5. Comparison of both approaches with respect to selected aspects of data generation.

State of the Art analysis

Software to build synthetic data

- SynGen: Synthetic Data Generation (https://ieeexplore.ieee.org/document/9697232)
- GenEthos: A Synthetic Data Generation System With Bias Detection And Mitigation (https://ieeexplore.ieee.org/document/9885653)

Software for sensor simulation (large scale) (https://www.technia.de/simulation/)

Forcasting of data

- Facebooks prophet (https://research.facebook.com/blog/2017/2/prophet-forecasting-at-scale/)
- Demand Forecasting Using Artificial Neural Network Based on Quantitative and Qualitative Data (https://ieeexplore.ieee.org/document/9245614)
- Synthetic Test Data Generation Using Recurrent Neural Networks: A Position Paper (https://ieeexplore.ieee.org/document/8823801)

Software Requirements

Data Management:

1. Data Import: Ability to import data from various sources and formats (numpy, json, csv, …). ✓
2. Data Transformation: Features for transforming and preprocessing data (normalization, linear trend removal, etc). ✓
3. Data Visualization: Visualization tools to explore data and understand its distribution and characteristics. ✓

Software Requirements

Model Training:

1. Model Selection: A wide range of machine learning models to choose from.
2. Parameter Tuning: Tools for hyperparameter tuning and model optimization. ?
3. Cross-Validation: Built-in support for cross-validation to assess model performance.
4. Scalability: Ability to handle large datasets and scale computations as needed.
5. Performance Metrics: A variety of metrics to evaluate model performance (accuracy, precision, recall, F1 score, etc.).

Software Requirements

User Interface and Experience:

1. Ease of Use: An intuitive and user-friendly interface.
2. Documentation and Help: Comprehensive documentation and help resources.
3. Collaboration Tools: Features to enable collaboration among multiple users. (currently out of scope)
4. API Integration: Ability to integrate with other tools and services via APIs.

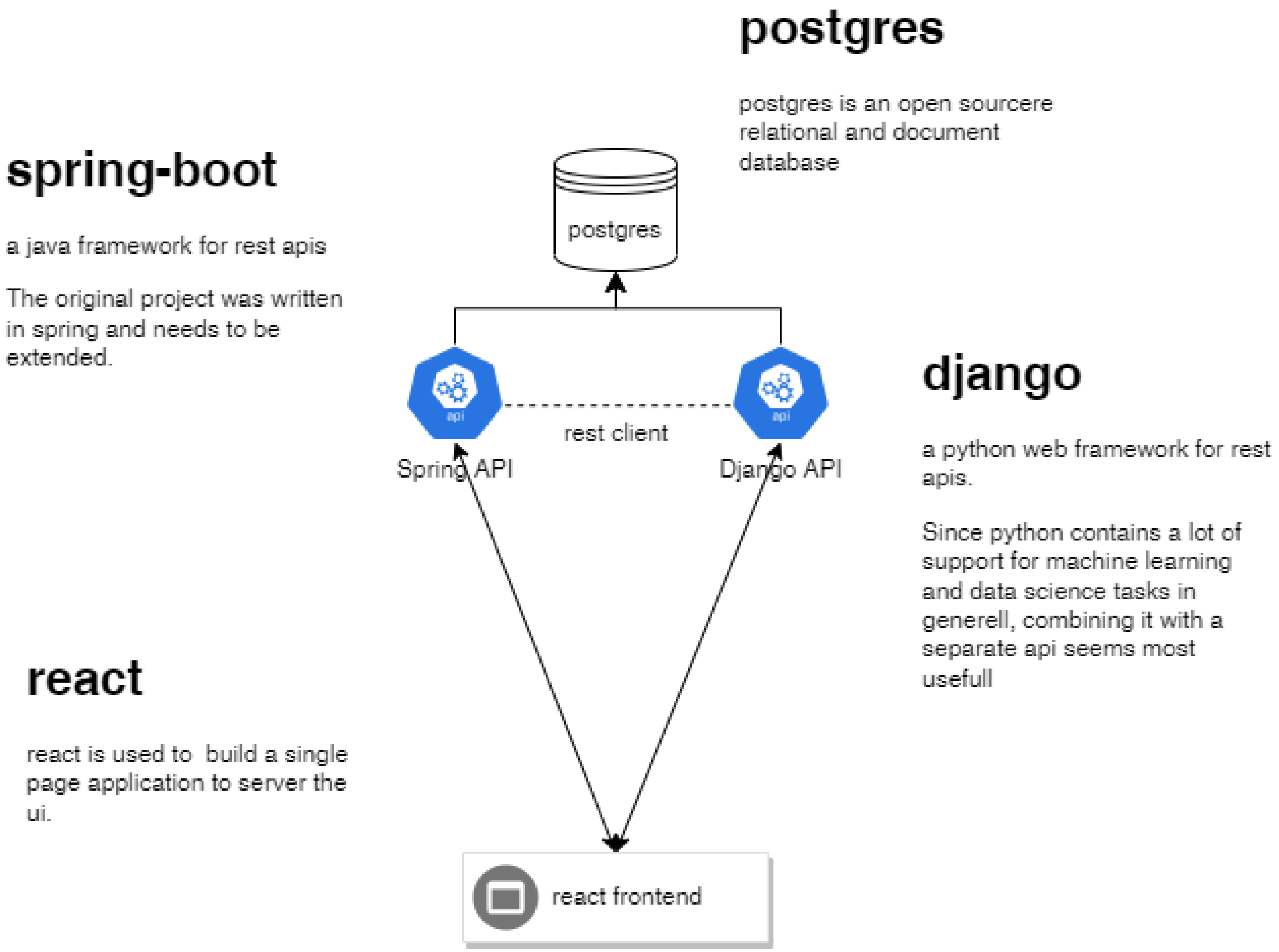
Software Requirements

Security and Privacy:

1. Data Security: Ensuring that user data is stored and processed securely. ✓
2. Privacy Compliance: Compliance with relevant data protection and privacy regulations. ✓
3. User Authentication: Secure user authentication mechanisms. ✓

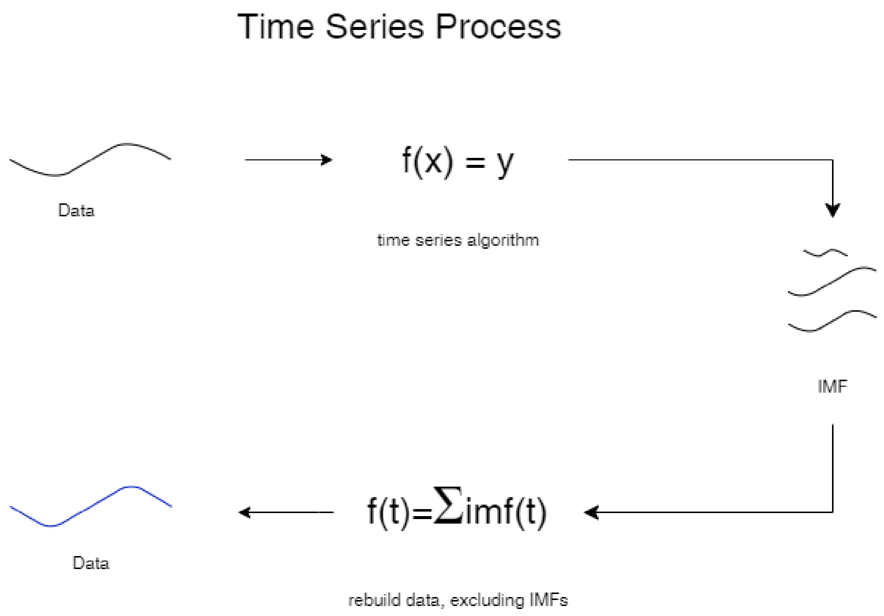
Solution Idea

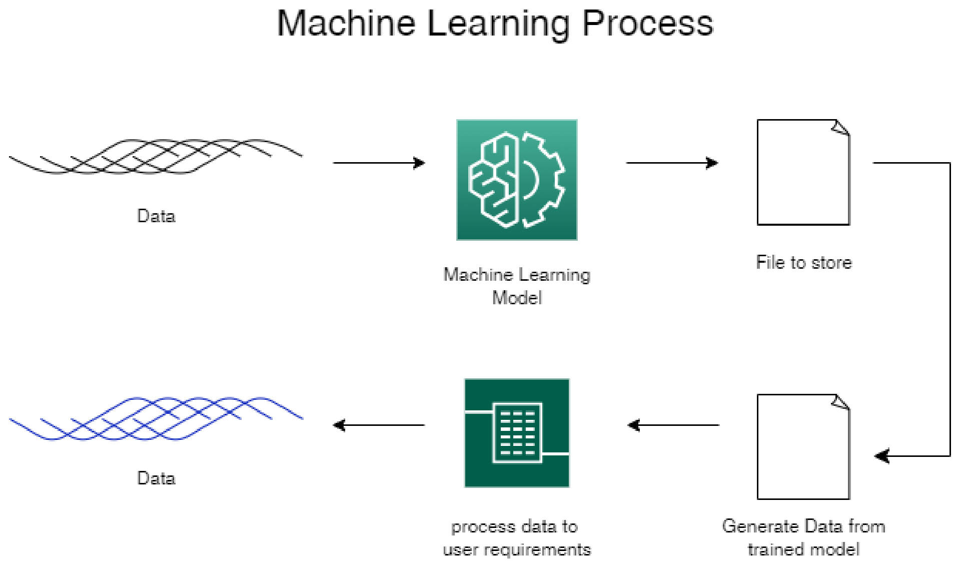
- Add additional API, used for training and generating synthetic data
- Integrate training process into frontend
- Integrate new API into existing data structure



Solution Idea

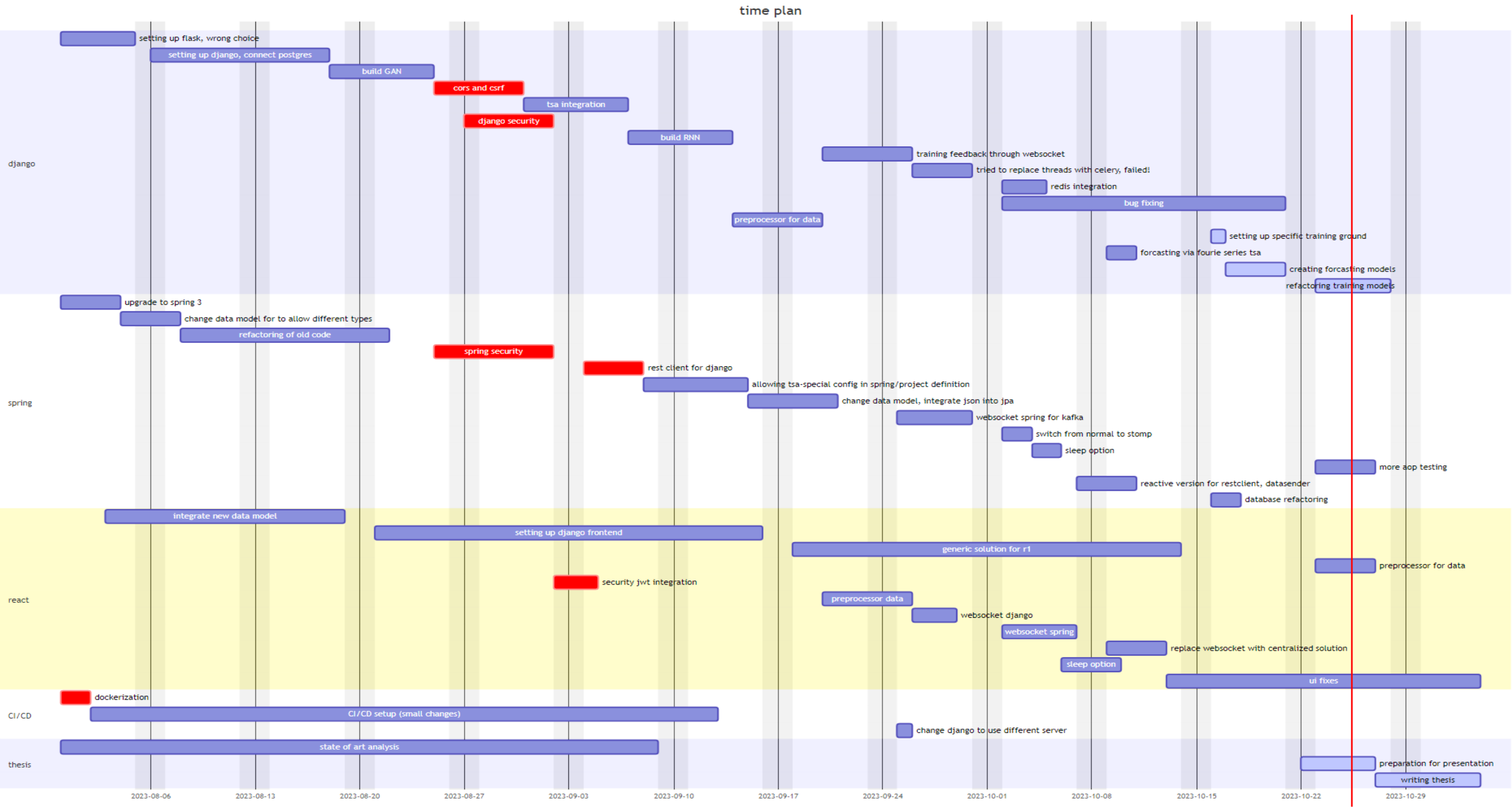
How to generate the data?





What has been done?

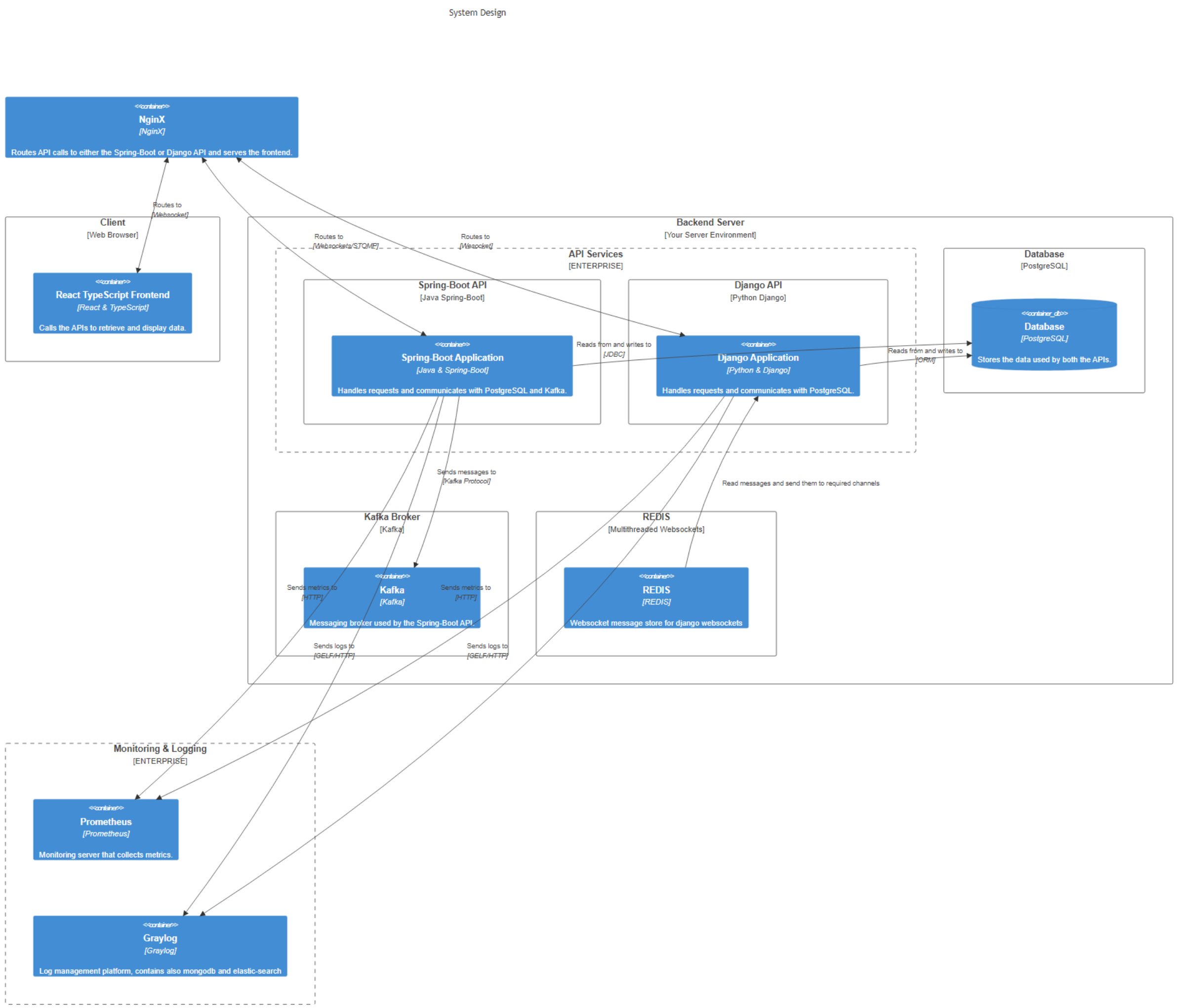
What has been done?



What has been done?

Current Solution Design

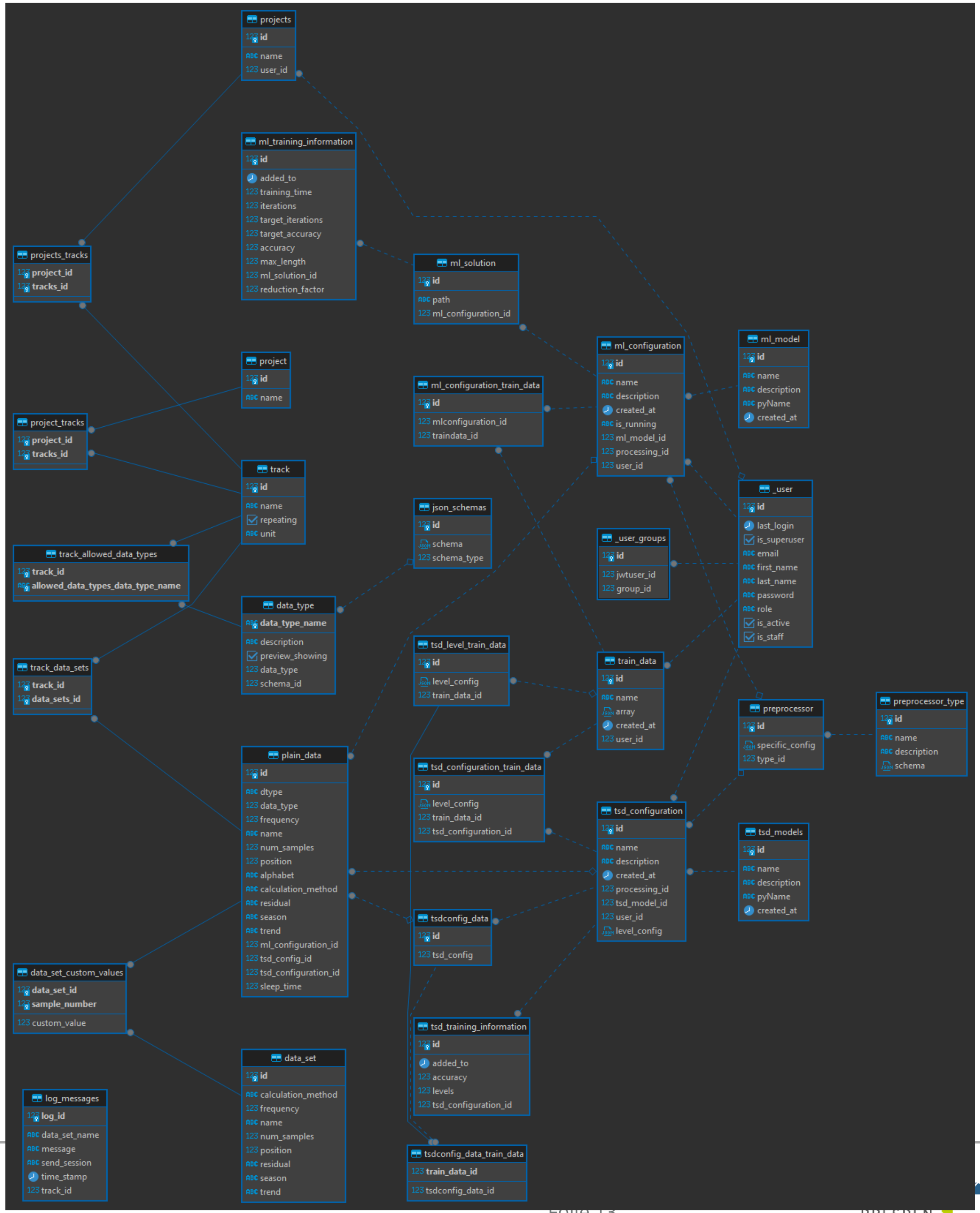
- Django application
- Spring boot application design change
- Prometheus and Graylog for centralized logging
- Redis for multiprocessing websockets
- nginx as reverse proxy (neccessary for hosting)
- Gitlab ci/cd for automatic deployment



What has been done?

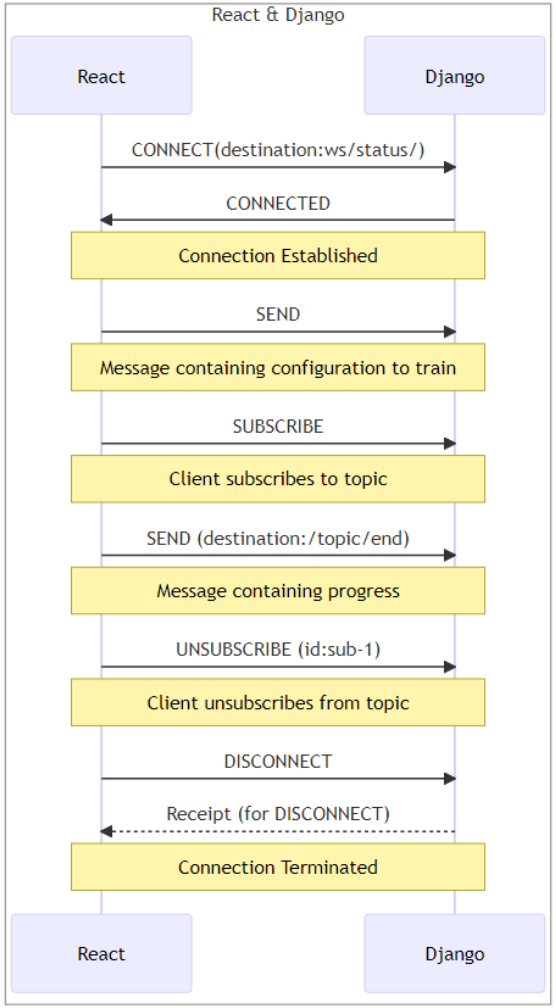
Current Solution Design

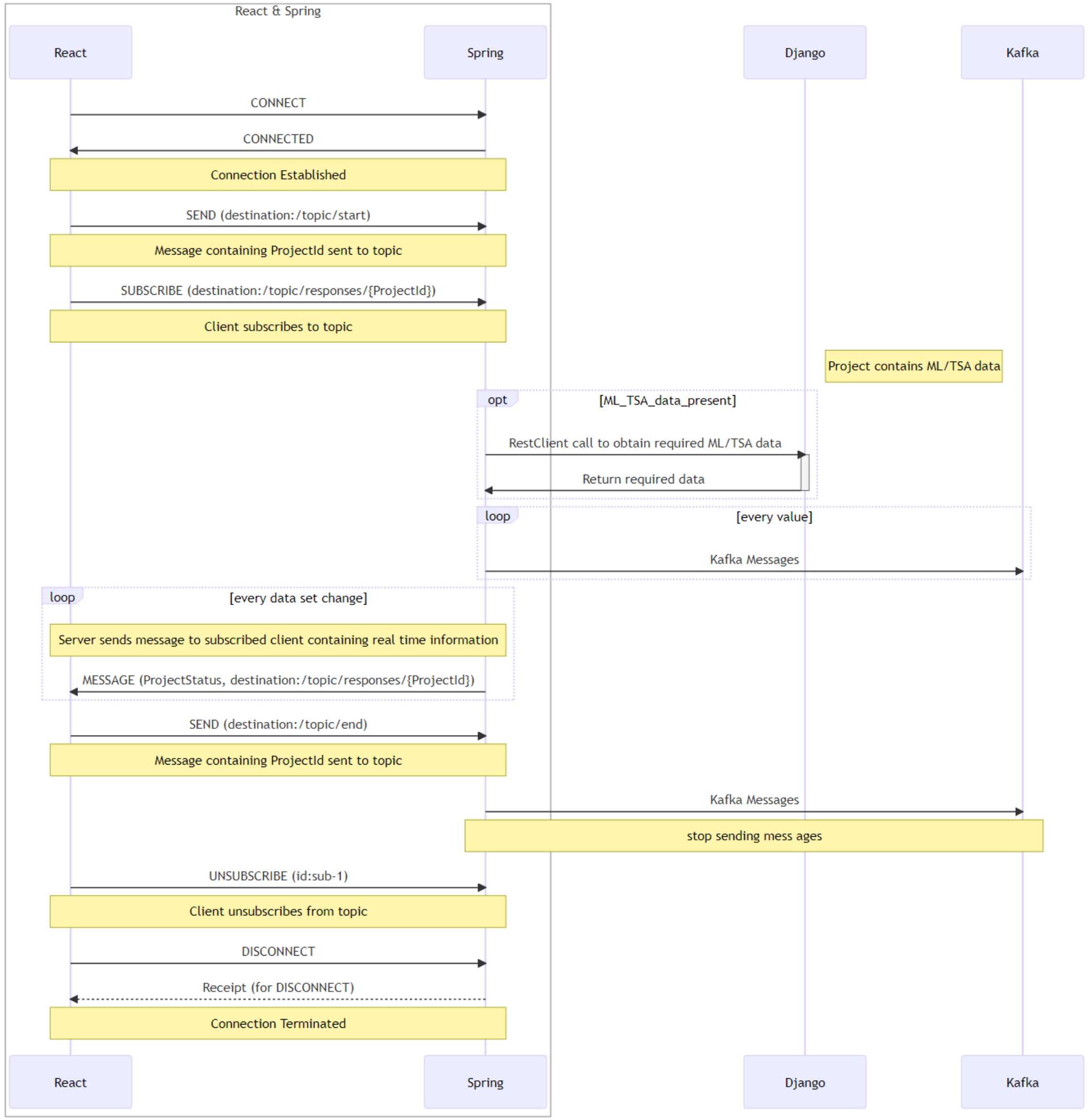
- Database Design has been extended
- User as centralized figure



What has been done?

Websocket connections can improve user feedback



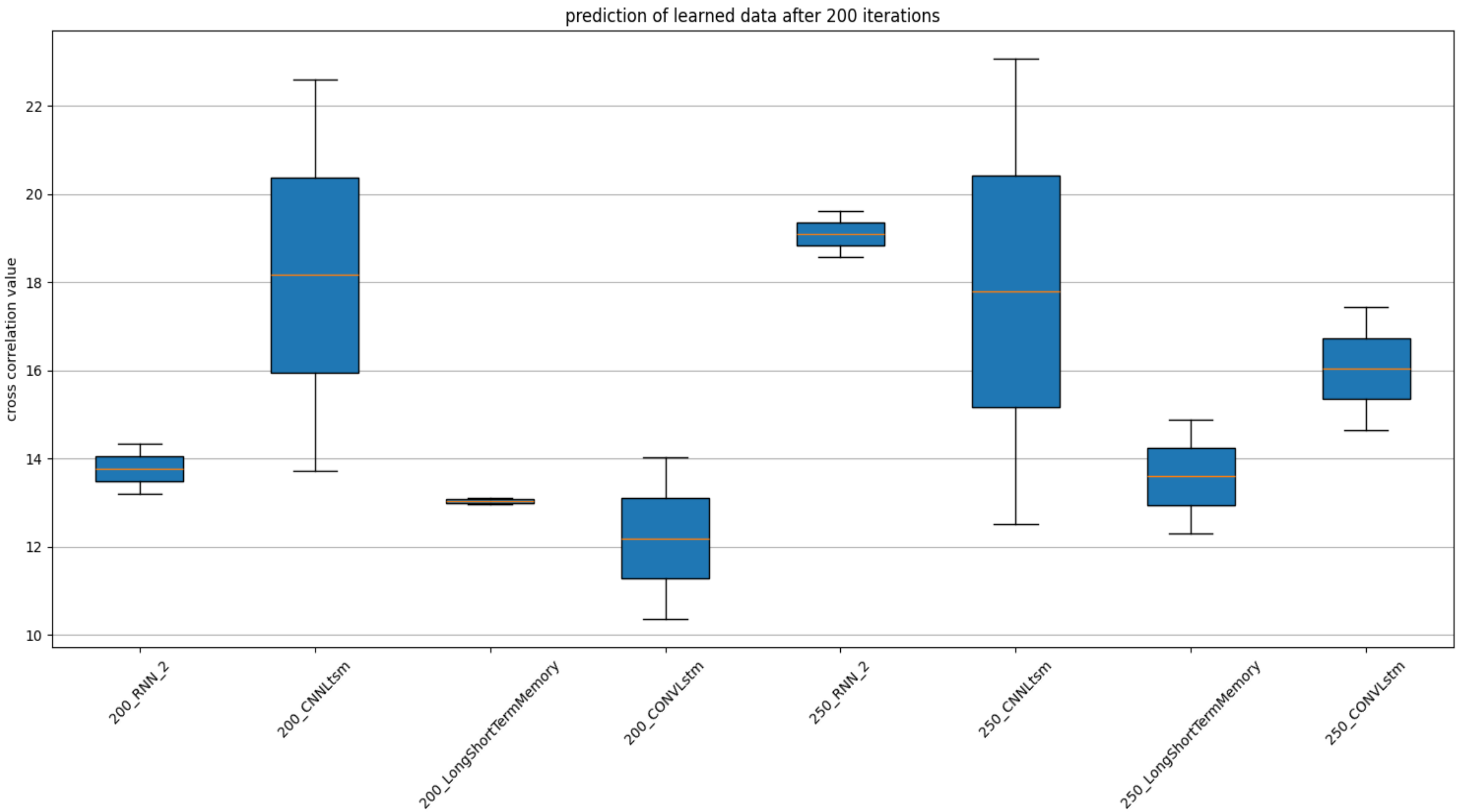


What has been done?

Machine Learning Models:

- GAN
- CGAN
- WGAN (work in progress)
- TGAN (work in progress)

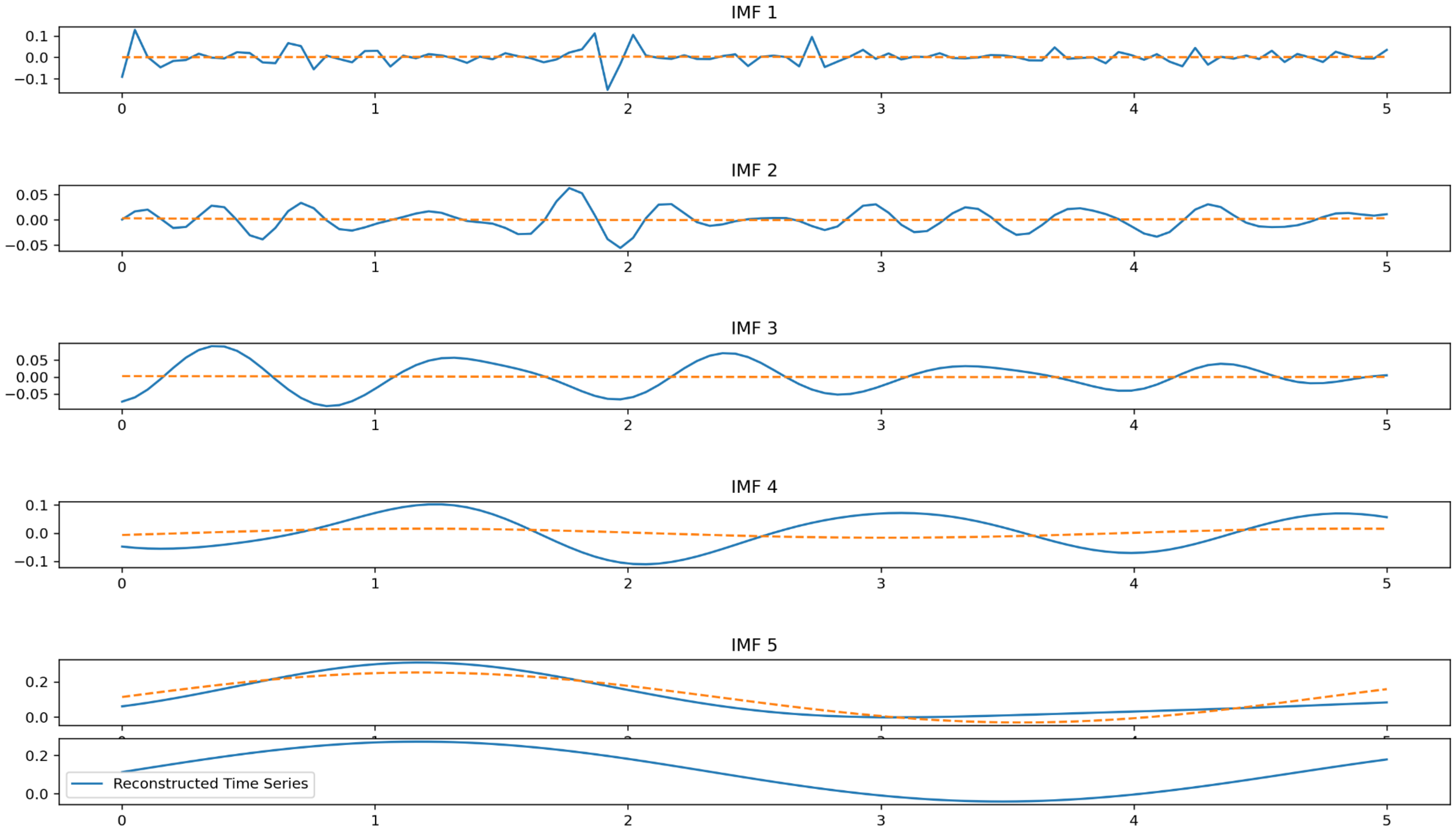
- LSTM
- RNN
- CNN



What has been done?

Time Series Analysis:

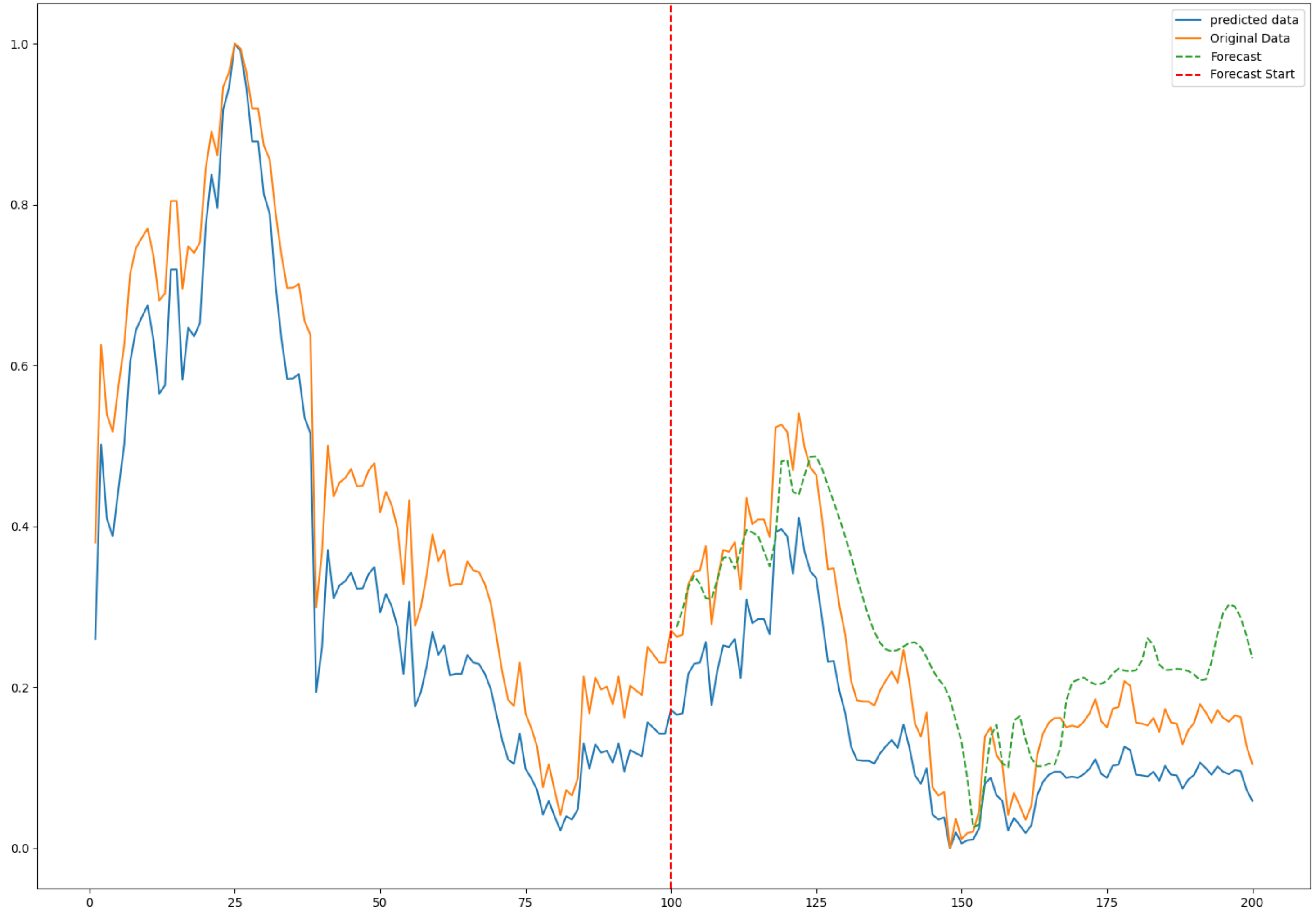
- EMD
- SSA



What has been done?

Long short-term memory model

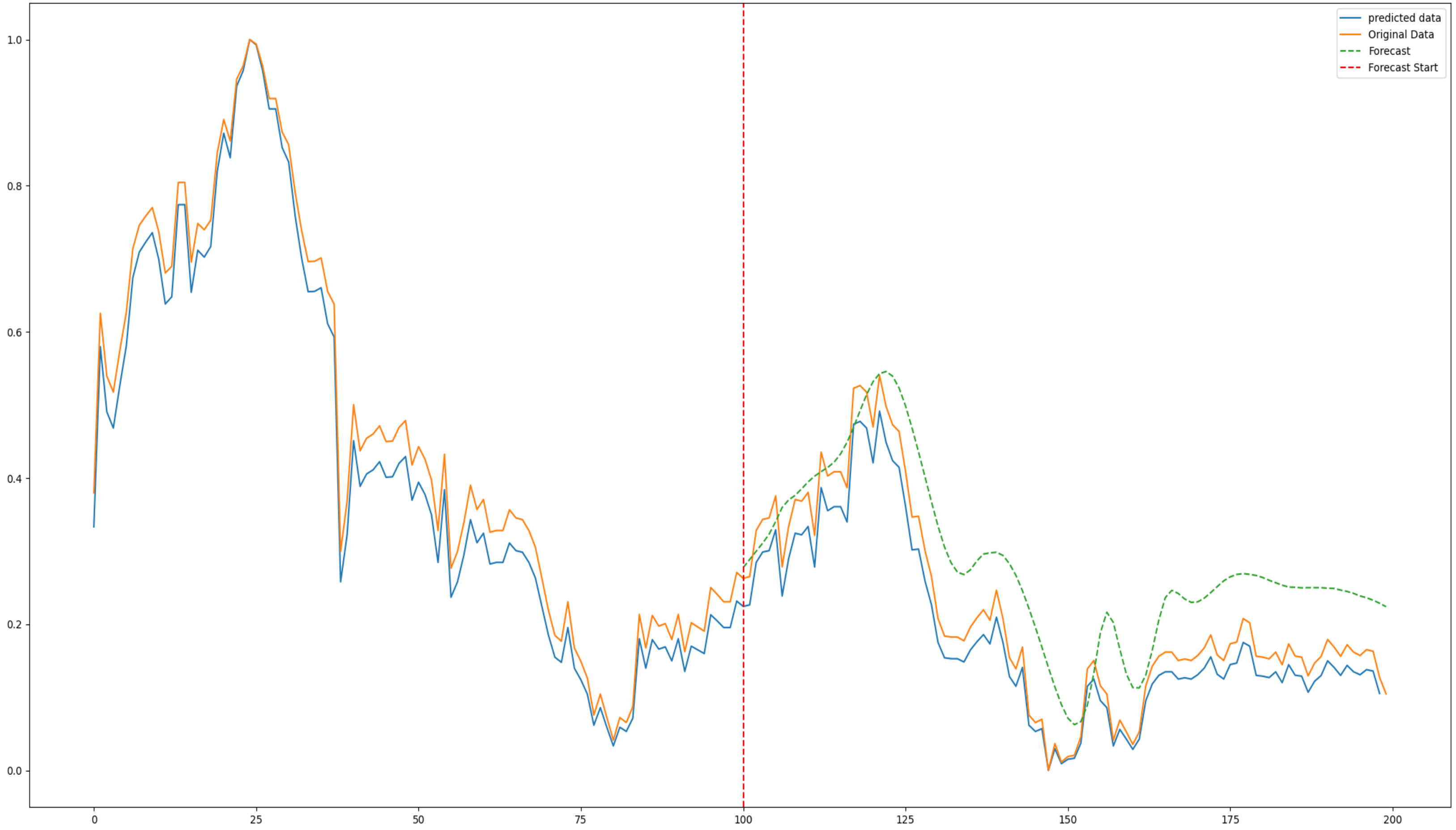
- 1000 iterations over 4 datasets
- Training time
  dell xps 13 plus: 14min
  Orange pi 5b: 160min



What has been done?

RNN

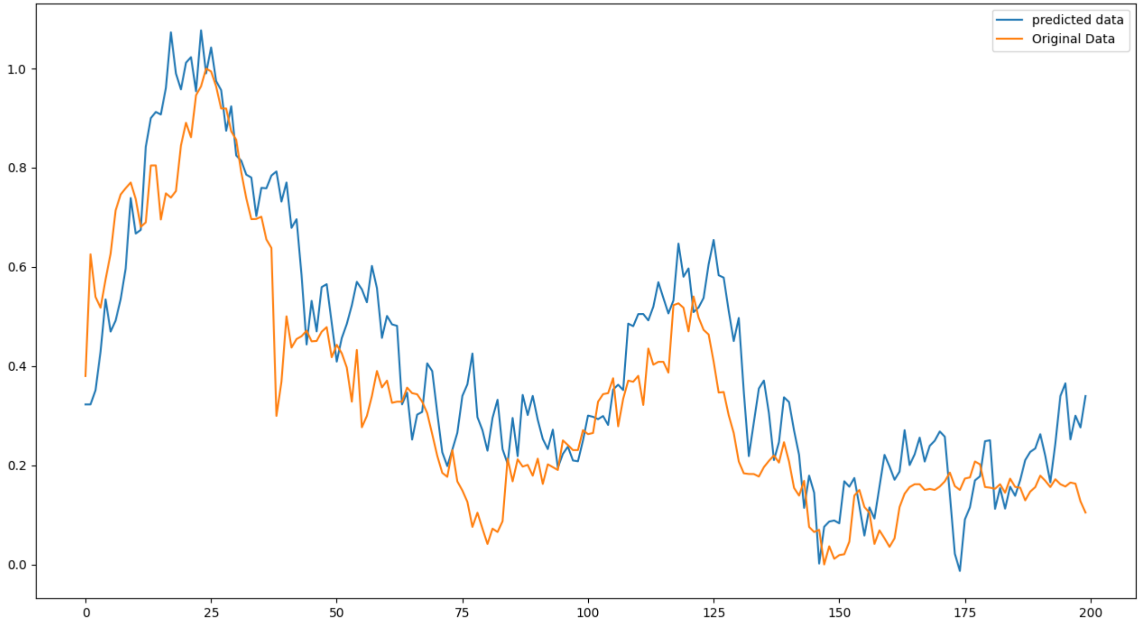
- 200 iterations over 4 datasets
- Training time:
  Orange pi 5b: 15min



What has been done?

GAN

- 1000 iterations over 4 datasets



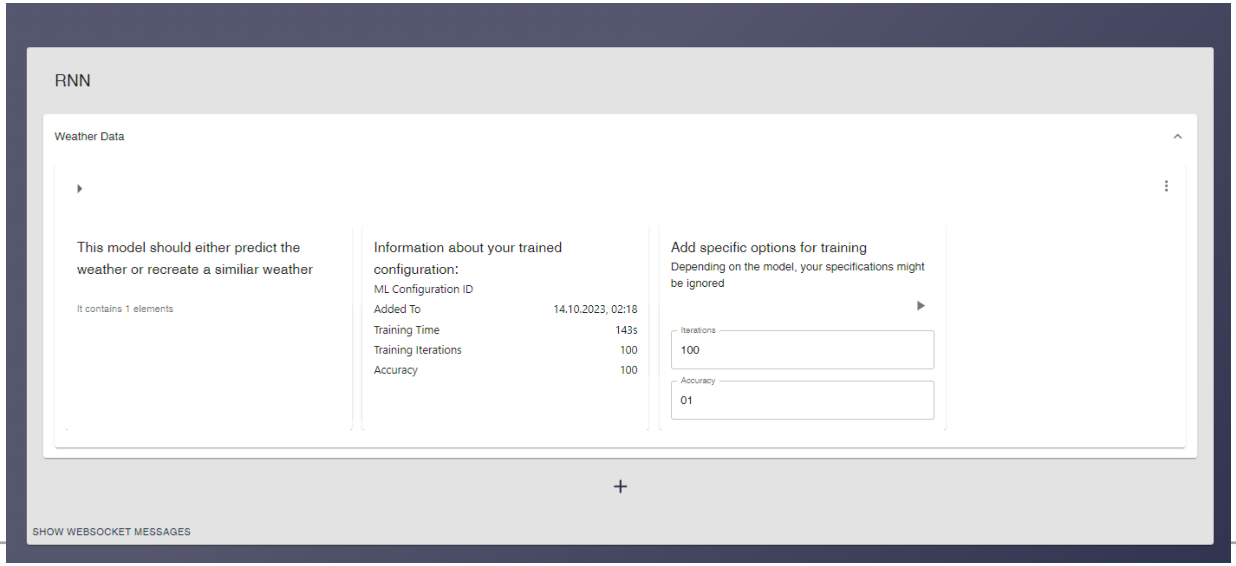
CGAN

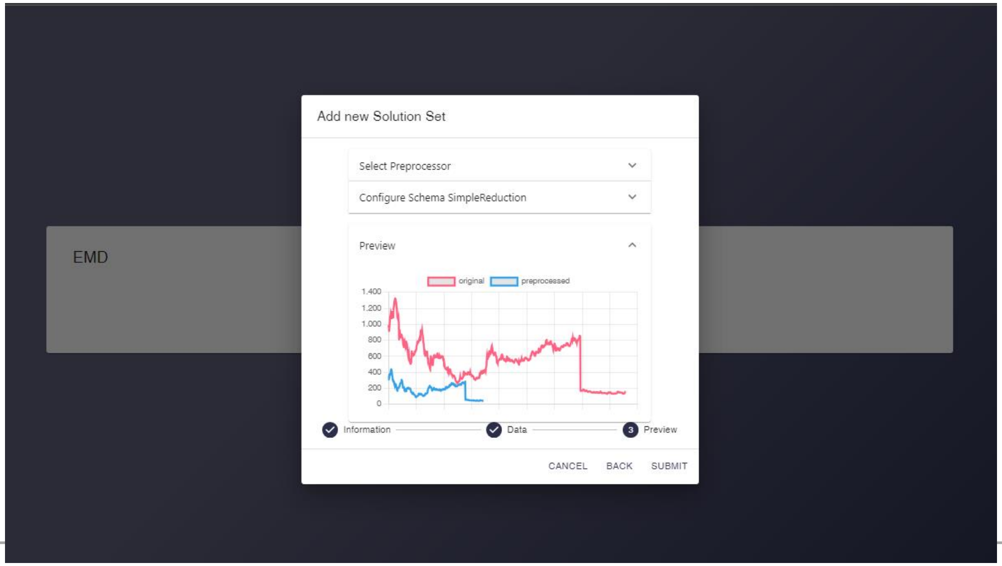
- 1000 iterations over 4 datasets

What has been done?

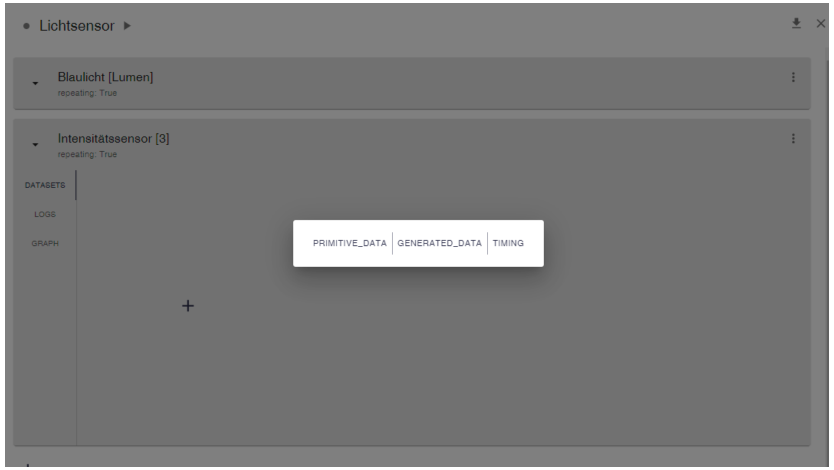
What has been done?

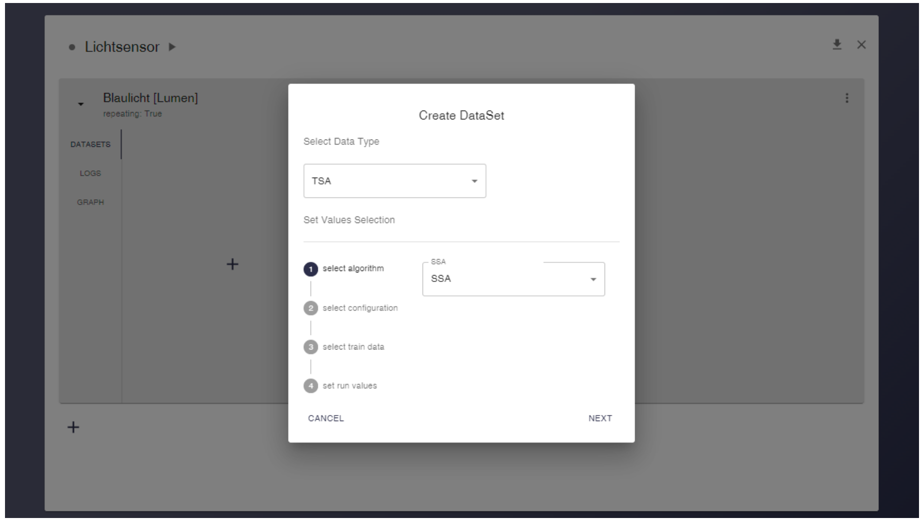
- Unified UI with specious forms for controlled inputs
- Visual feedback of used data
- Customization options

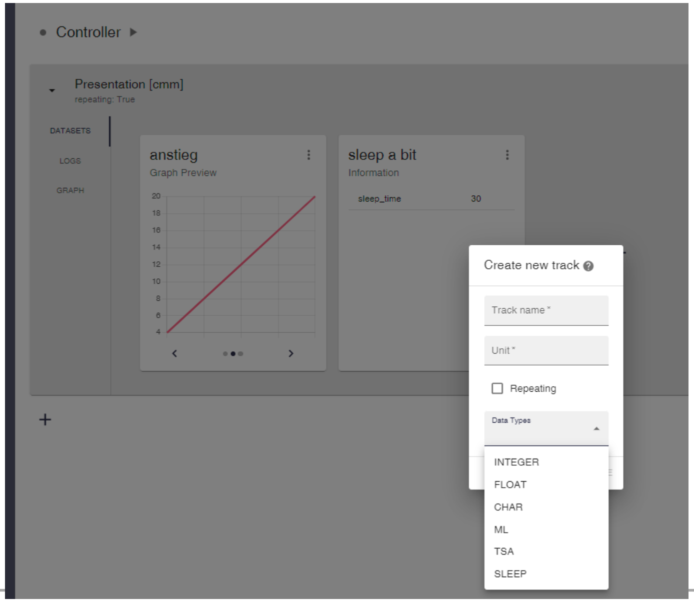


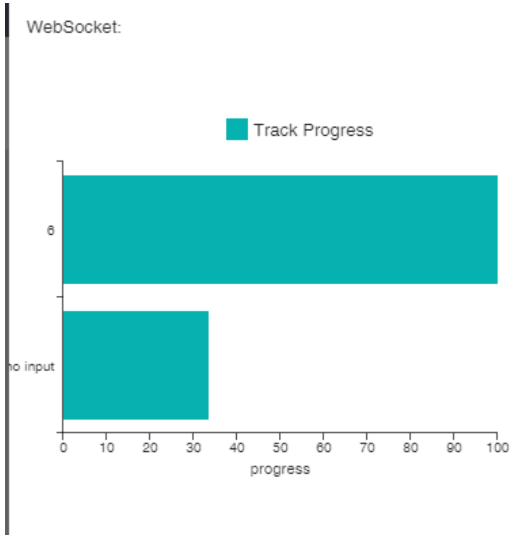


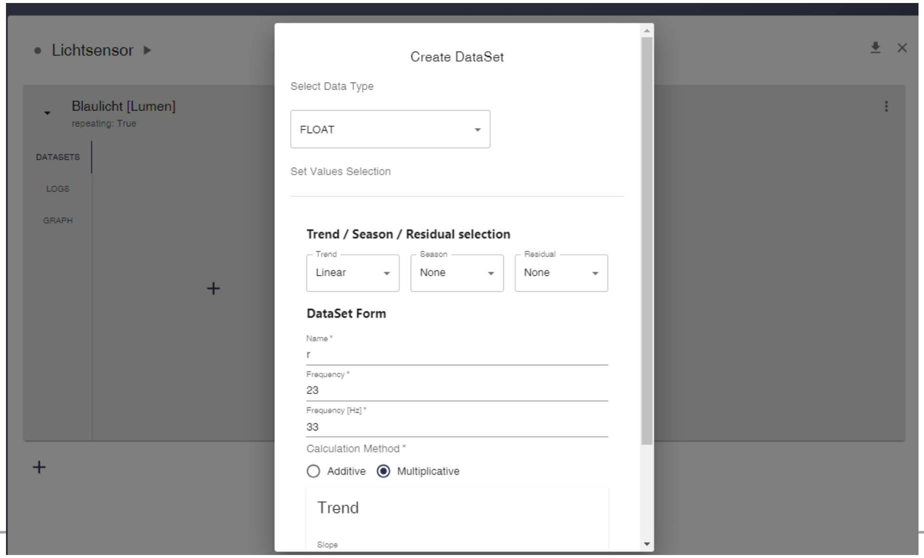
What has been done?









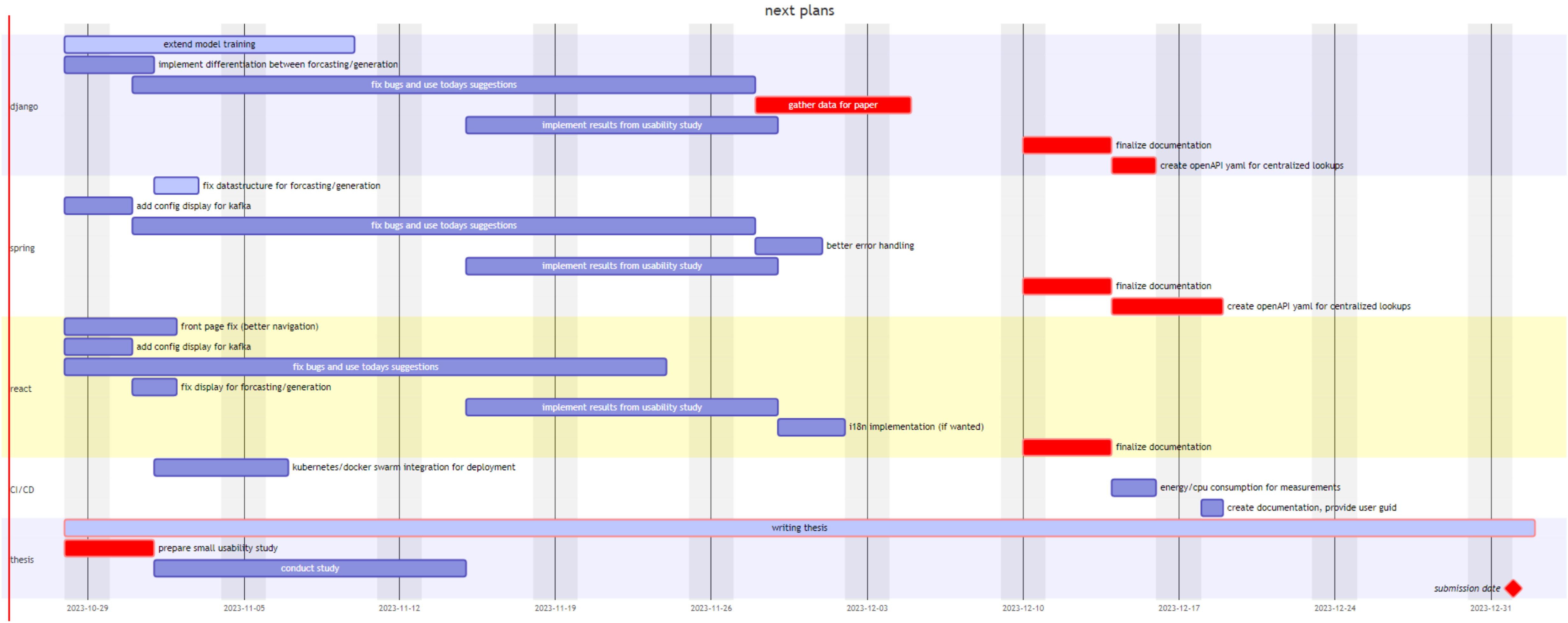


Current Status
What challenges have arisen?

Training of models:

- Current state of the art papers do not offer extensive descriptions/examples about their configuration (crucial for GANs)
- GANs are hard to train, with variing data, they can easily converge.
- Data shape is unknown, training a model specifically for one kind is not possible/unadvisable
- Shared security (keycloak would be an option here)
- Websockets (basic websocket vs stomp)
- Energy measurements for python code exists, but dont work on existing hardware -> measure cpu usage of docker container and exclude

Time Schedule and next steps



Video/Demo

Questions

UI/UX Improvement

Enhancing UI/UX: How can we better showcase the benefits of forecasting and synthetic data generation in our user interface?

Model Training

Balancing User Control: What's the right level of user control in model training to balance between achieving optimal results and preventing excessive training attempts?

Project Sharing

Facilitating Collaboration: Requires database and permission changes to enable project sharing, and how might this improve collaboration?

Mathematical Comparisons

Expanding Analysis Options: Beyond correlation, what other mathematical comparison methods should be integrated for more comprehensive analysis?

Forecasting Models

Diverse Forecasting Options: Is there an interest in forecasting models, like Prophet, to provide our users with more choices?

Integration with External Models

Accessing External Models: Services like AWS Lambda could be used for other machine learning models from y_data, since it clashes with current data?

Models
Generative Adversarial Networks

- Structure: Comprises two neural networks, the generator and the discriminator, which are trained simultaneously through adversarial training.
- Function: The generator creates data, while the discriminator evaluates it.
- Objective: The generator aims to produce data so realistic that the discriminator cannot differentiate it from real data.
- Applications: Image generation, style transfer, image-to-image translation, and more.
- Challenges: Mode collapse, training instability, and the need for a balance between the generator and discriminator.

Models
Time-series Generative Adversarial Networks

- Specialization: Designed specifically for generating realistic time-series data.
- Structure: Similar to GANs but incorporates modifications to handle the temporal nature of the data.
- Applications: Financial data simulation, network security, and any domain requiring synthetic time-series data.
- Advantages: Helps in augmenting time-series datasets, which are often limited in size.
- Challenges: Requires careful design to capture temporal dependencies and ensure consistency over time.

Models
Conditional Generative Adversarial Networks

- Extension of GANs: Introduces additional conditional variables to the generator and discriminator, providing control over the generated data.
- Function: Allows the generation of data with specific characteristics or classes.
- Applications: Image-to-image translation, style transfer with specific attributes, and controlled data generation.
- Advantages: Provides a way to direct the data generation process, making it more versatile.
- Challenges: Requires careful selection and integration of conditional variables to ensure effective control.

Models
Wasserstein Generative Adversarial Networks

- Improvement over GANs: Addresses training instability and mode collapse issues in traditional GANs.
- Key Change: Utilizes the Wasserstein distance as the loss function, providing a more stable and meaningful gradient for training.
- Advantages: Leads to more stable training and helps in generating more diverse samples.
- Applications: Image generation, style transfer, and any application benefiting from stable GAN training.
- Challenges: Computationally more expensive due to the need for weight clipping or gradient penalty to enforce the Lipschitz constraint.

Models
Recurrent Neural Networks

- Structure: Networks with loops to allow information persistence, where previous outputs are used as inputs for the next step.
- Function: Designed to work with sequence data, capturing temporal dependencies.
- Applications: Text generation, machine translation, speech recognition, and any task involving sequence data.
- Advantages: Can process sequences of variable length and capture temporal dependencies.
- Challenges: Susceptible to the vanishing gradient problem, making it difficult to capture long-term dependencies without modifications like LSTM or GRU cells.

Models
Convolutional Neural Networks

- Structure: Composed of convolutional layers, pooling layers, and fully connected layers.
- Function: Primarily used for grid-like data such as images, where the convolutional layers act as feature detectors.
- Applications: Image classification, object detection, image generation, and any task involving grid-like data.
- Advantages: Parameter sharing and local connectivity make CNNs efficient and effective for image-related tasks.
- Challenges: Requires a large amount of labeled data for training, and the architecture needs to be carefully designed for each specific task.

Models
Long Short-Term Memory networks

- Structure: A type of RNN with special units called memory cells and three types of gates (input, forget, and output) to regulate the flow of information.
- Function: Designed to capture long-term dependencies and avoid the vanishing gradient problem common in traditional RNNs.
- Applications: Time-series prediction, natural language processing, speech recognition, and any task requiring the understanding of long-term dependencies.
- Advantages: Capable of learning and remembering over long sequences and is less susceptible to the vanishing gradient problem.
- Challenges: More complex and computationally expensive than simple RNNs, and can be prone to overfitting on smaller datasets.

Models TSA
Empirical Mode Decomposition

- Structure: A data-driven method that decomposes a signal into a set of intrinsic mode functions (IMFs) and a residue, without requiring a priori basis functions.
- Function: Aims to identify and separate oscillatory modes inherent in the data, capturing both amplitude and frequency modulations.
- Applications: Signal processing, time-series analysis, biomedical signal analysis, and any field requiring adaptive time-frequency analysis.
- Advantages: Adaptive and suitable for non-linear and non-stationary data, capturing intrinsic oscillations effectively.
- Challenges: Mode mixing (where a single IMF contains signals of widely disparate scales) and end effects (distortions at the signal boundaries).

Models TSA
Singular Spectrum Analysis

- Structure: A non-parametric spectral estimation method that decomposes a time series into a sum of interpretable components such as trend, oscillatory patterns, and noise.
- Function: Extracts information from short and noisy time series, identifying underlying structures without prior assumptions about the data.
- Applications: Climate data analysis, economic time series, signal processing, and any application requiring trend and pattern extraction from time series.
- Advantages: Requires minimal prior knowledge about the data, and can separate signal components effectively even in noisy settings.
- Challenges: The choice of window length is crucial and can affect the results significantly; interpreting the decomposed components requires expertise.