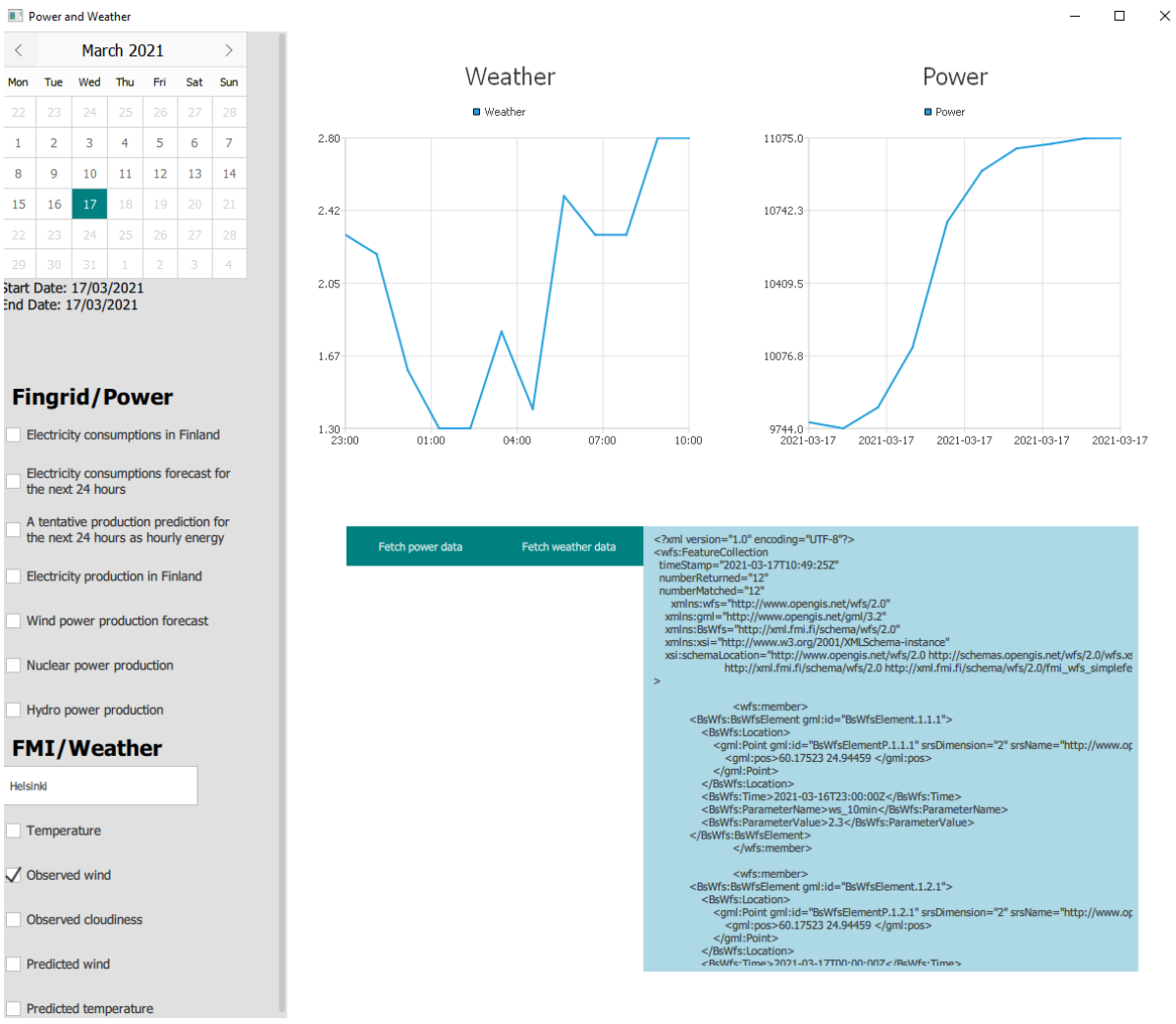
PowerWeatherApp

Dependencies

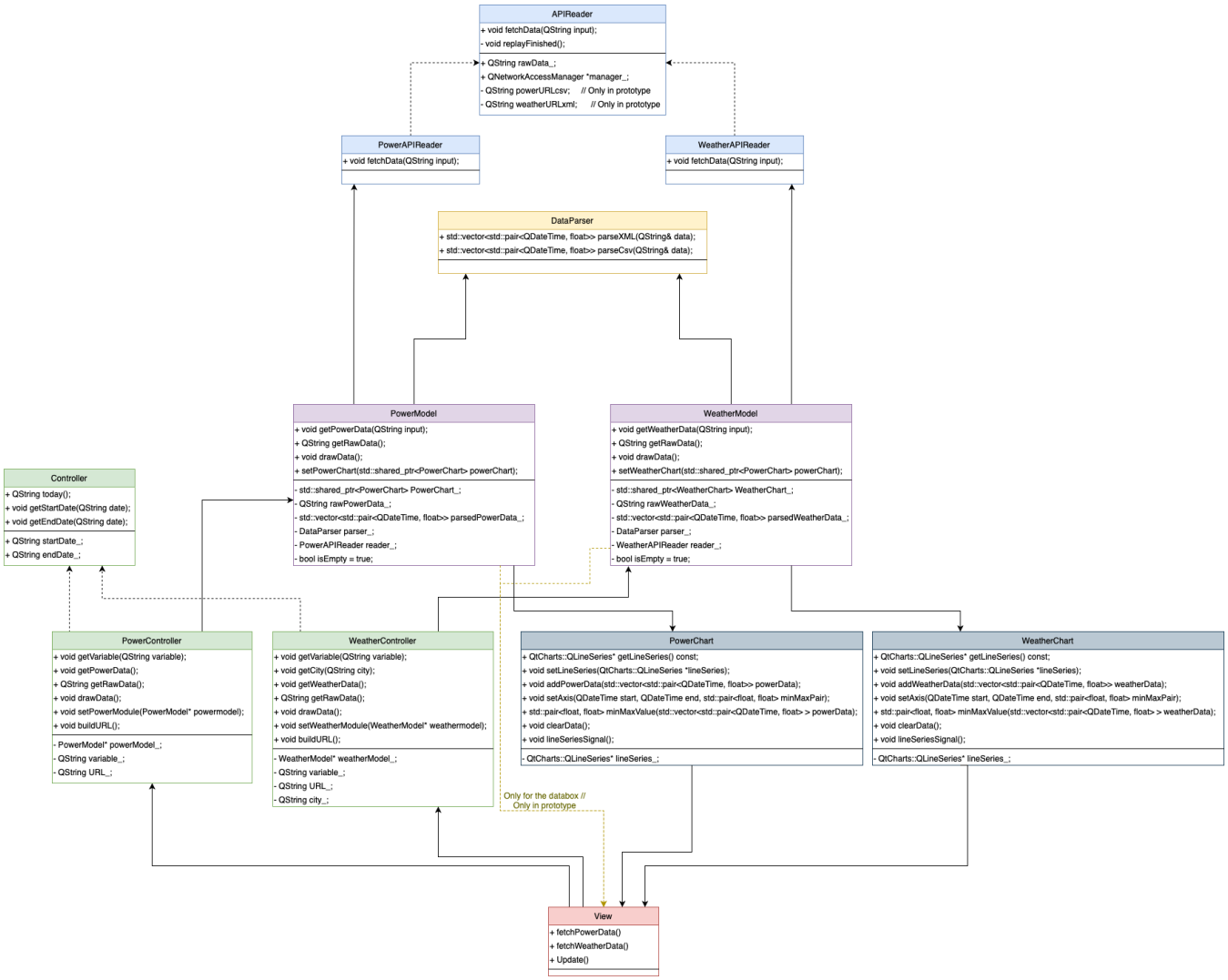
- Using QT 5.15.2 (MSVC 2019), minGW 8.1.x
- QT: quick, widgets, network, quick charts (.pro file includes)
- OpenSSL 1.1.1d or newer

Design

- Using two different controllers for power (Fingrid) and weather (FMI) which are inherited from parent controller.
- We're using semi-active MVC (pull model).
- Using QML objects for viewing relevant data according to the user's preferences.
- Backend for requesting API information is done by using C++.
- Frontend is done using qml.



Class diagram



Components

- User is able to choose what kind of data he/she wants to get in the view component.
- User's preferences go to the controllers and the URL is build based on those preferences. The preferences are then sent into the models.
- Models read data with APIReaders and parse said data with data parser. Models also hold the data.
- Parsed data is displayed with the charts in the view.

Decision-making process

Prototype (DONE)

- Right now, we're thinking of making two different models for the different APIs because the data differs too much, and we couldn't find a proper relation between the two APIs. In the future if necessary, we separate controller from the model classes. (DONE)
- Both APIs need to be parsed separately because we get power as csv format and weather as xml format. (DONE)

Midterm

- Based on prototype phase meeting we decided to change our class diagram to more efficient solution (inheritance). We also changed the way how data flows between classes (through controllers).
- We inherited controllers from a parent controller and we're thinking of doing the same thing to models and charts in some point.
- UI databox will probably be changed into an instruction box. Its functionality is telling the user some feedback of the choices made.
- We didn't separate the parsers because they only have two functions. Their responsibilities are to only parse raw data.

Todo

- We still need to implement some functions into FMI/Weather (observed cloudiness, predicted wind, predicted temperature).
- We will add a feature to make the view component able to give feedback to the user according to the preferences.
- We are planning that we could save the chart in .png format and save the data in .txt file so you could load the data without online connection.
- Commenting the code better and making it overall more readable.

Self-evaluation

- After prototype phase meeting our UI has not changed. We are happy with the current UI.
- We were able to stick to our own original design. We have added many classes so the code is easier to edit.
- We think our code is more than 30 % ready. We are ahead of our original work schedule.
- We have split the workload pretty evenly and we have all agreed that everybody has participated in the making.
- Most of the productive coding has been done via Discord meeting where everybody in the group attended.