Iteration one
In the first iteration, I planned how the dashboard could work. I outlined the following parts:

- A page menu to navigate between the different pages, temporarily set as QTableViews until the pages are fully developed.
- A language menu to select a language.
- Pollutant cards with labels for the type of pollutant.
- Filters for location, month, and day to showcase data from specific locations and specific dates.
- A feature to load a CSV file.

First, I started by developing the page menu to provide navigation between the pages. The page menu is a dropdown menu in the menu bar, displaying all available pages. Pressing one of the options changes the central widget to one of the temporary QTableViews. Initially, I struggled with connecting the buttons to a set-page function that changes the central widget. I encountered an issue with passing arguments, so I used separate functions to set the central widget.

Next, I started working on the cards. The cards are linked to their respective pages and function as buttons to change the central widget, similar to the page menu. I stored the cards in a toolbar on the left side.

To load the CSV file, with the help of others in my group, we adapted the solution from Coursework 2 to handle CSV file loading. We then developed a method to filter the data based on location using QSortFilterProxyModel().

Iteration two
Next, I started working on a way to filter by date and location. To enable filtering by location, month, and day, I researched how QSortFilterProxyModel works and created a subclass of it. This subclass takes location, month, and day values as input and filters through each row of the CSV file to display the relevant data.

After the filter was working, I revisited the page selector to rework the connections. With some help from my group members, i managed to pass arguments to the setPage function via the connect method.

Afterwards, I started working on the translation feature. I struggled a lot with incorporating this into the program, as I was unable to get it to compile despite following the instructions in the slides and other sources. In the meantime, I added a language selector to the menu bar with English and Swedish as options, Swedish being my first language.

Iteration three

For iteration three, I helped complete some unfinished pages, focusing primarily on the pollutant overview page. The overview page consists of a search bar and a bar chart. When you input the name of a pollutant, the average value for that pollutant is displayed for each month of the year. The values are added to the chart through a method called updateChart. In this method, the string entered in the search bar is used to gather all measurements of the pollutant. The minimum and maximum average values are identified and used to set the scale for the y-axis.

Per the project requirements, the bars should be displayed in different colours depending on whether the pollutant is within a certain level. However, implementing this functionality posed a significant challenge due to Qt's chart syntax. One QBarSet and one QBarSeries were defined. All pollutant values are added to the QBarSet, which is then added to the QBarSeries. The issue was that only one colour could be applied to the QBarSet. Creating a separate QBarSet for each value and appending these 12 QBarSets to the series resulted in a problem where all bars appeared over January in the chart.

After re-evaluating our interpretation of the pollutant cards, I instead developed a main menu that displays these cards as large buttons leading to the different pages. The main menu is the first thing a user sees when launching the application. Additionally, there is a main menu button at the top of the toolbar.

Implementing this main menu was very simple. I created a subclass of QWidget containing four large, vertically stacked buttons. These buttons emit a signal, buttonClicked(const QString& page), which is then connected to the setPage function in the main window class.

For my next task, I integrated the Compliance page, developed by another group member, into the main window. After making some minor adjustments, the page was fully functional.

To enhance the presentation of our application, I developed a stylesheet that included several components used throughout the application. I chose a greyish colour palette for a clean, modern design, with a blue tint to emphasise the application's purpose of presenting water-quality statistics.