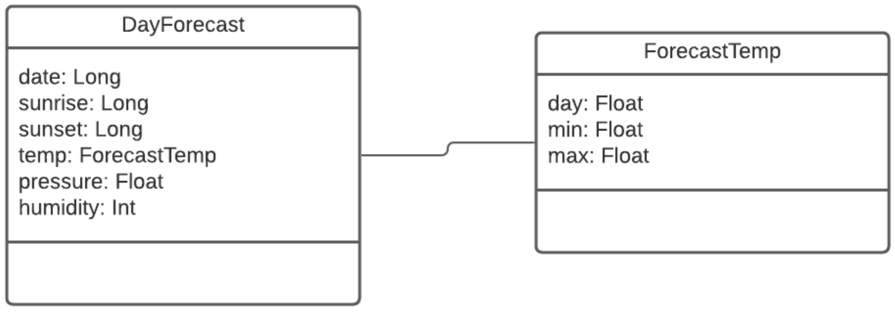
# Assignment 3

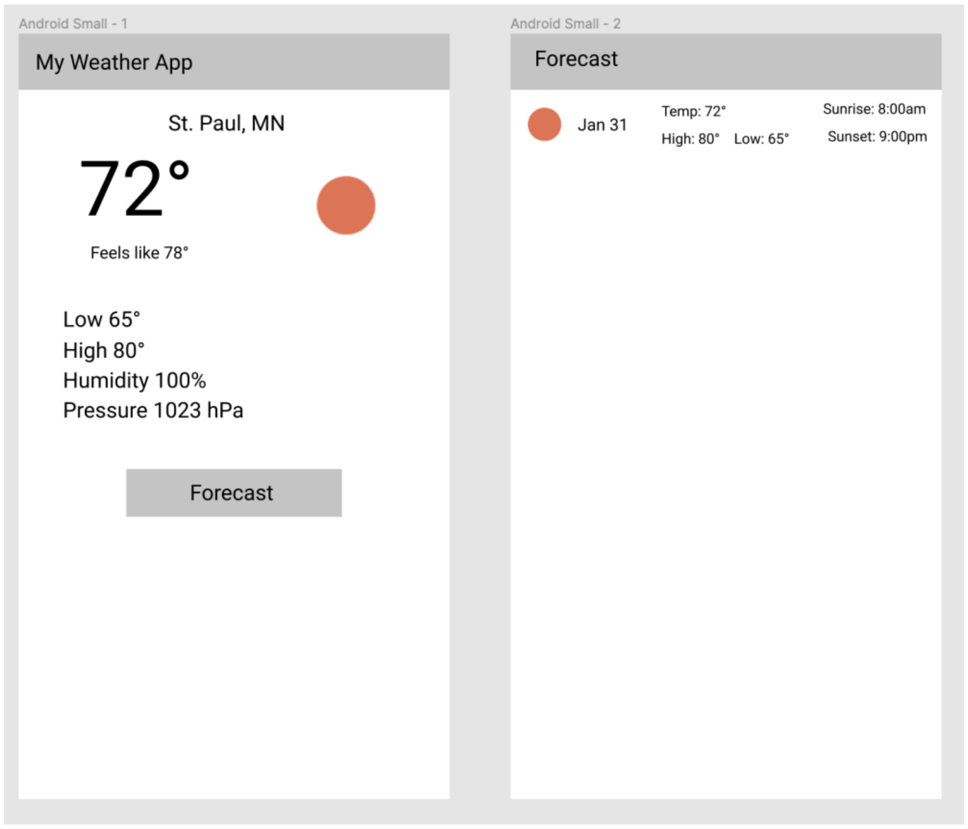
**40 points**

**Due: September 28, 2022**

In this assignment you will create a RecyclerView to display forecast data. For now, all data will be hard coded. You will add a button to the current conditions screen. When the user taps on the button, navigate to the Forecast Fragment. On the Forecast Fragment, display the forecast data as a list of items.

1. Add the RecyclerView dependency to your module level build.gradle file.
2. Add the AndroidX Navigation libraries to your module level build.gradle file.
3. Refactor your Main Activity to a Main Activity with a Current Conditions Fragment which shows the same user interface you developed in Assignment 2. For hints on how to do this, review the Week 3 lecture video.
4. Create a nav_graph.xml navigation graph.
5. Add a button to the current conditions screen labeled “Forecast.”
6. Create Kotlin data classes according to the object diagram below.
7. Create a ForecastFragment
8. Implement a RecyclerView and RecyclerView.Adapter to display forecast items as seen in the design flat below
9. Create dummy data in a List in the ForecastFragment. You must have at 16 items in your list. The data for each item should be different.
10. Use the dummy data you create in your RecyclerView.Adapter implementation.





Requirements:

- When the user taps on "Forecast" button, the ForecastFragment must be displayed.
- The ForecastFragment must display 16 items.
- When the user taps the back button, the ForecastFragment must exit, and the application should display the current conditions fragment.
- The application must compile and run without modification. If it doesn't compile and run it cannot be graded and you will receive a 0.
- Create a feature branch for your code.
- When done implementing features, commit your code to your feature branch, and push the code to GitHub.
- Create a PR in Github and submit the link to the PR in D2L.

Hints:

- Use a UTC calculator such as https://www.timestamp-converter.com to get the UTC timestamps for future dates and times
- Use the DateTimeFormatter class to get the display text which corresponds to the UTC Long timestamp
- Use LocalDate and LocalDateTime classes
- Start small.
    - Show a blank Forecast Fragment when the user taps the Forecast button

    - Create your data classes.
    - Get the RecyclerView to display the correct number of views based on the Adapter's backing data
    - Next focus on binding each view to the data object.
- Search on StackOverflow if you have questions.
- Read sample code

Point breakdown

- 30 – All requirements met
- 5 – Good code style and formatting
- 5 – Proper use of Git branches and GitHub PRs

Git Instructions

- Make sure you have merged your assignment2 branch into your main branch on Github.
- Checkout the main branch: "git checkout main"
- Pull the merged main branch from Github: "git pull"
- Verify that the code you expect to be there is ther
- Create a new branch for Assignment 3: "git checkout –b assignment3"
- Implement Assignment 3
- Add your changes to the staging area: "git add ."
- Commit your changes: "git commit –m "your git commit message""
- Push your changes: "git push origin assignment3"
- Go to Github and create your PR.
- Double check that all the code that you expect to be included is part of your PR

Resources

- RecyclerView - https://developer.android.com/guide/topics/ui/layout/recyclerview https://developer.android.com/jetpack/androidx/releases/recyclerview
- LocalDate - https://docs.oracle.com/javase/8/docs/api/java/time/LocalDate.html
- LocalDateTime - https://docs.oracle.com/javase/8/docs/api/java/time/LocalDateTime.html
- DateTimeFormatter - https://docs.oracle.com/javase/8/docs/api/java/time/format/DateTimeFormatter.html
- Instant - https://docs.oracle.com/javase/8/docs/api/java/time/Instant.html
- Create a LocalDateTime object from a timestamp - https://stackoverflow.com/a/44883570