**Backlog**

**Epics:**

* Back end
  + Configuration of data sources (e.g. yr.no)
    - As an admin, I want to set up my own data sources, so that I can configure my system
      * Set up configuration file for data sources (eg [www.yr.no](http://www.yr.no), credentials)
      * Create HttpClient for Yr to communicate (by using openAPI contract)
      * Set up authentication for httpClient (please use httpClientFactory)
    - As a system, I need a data model for storing data
      * Draw.io to draw database
      * Code first data model (entity framework, EFC) (datamodel context, push changes to database)
      * Create a dataservice that lets us write data to database
  + API feature (Get, Post, Put, Patch, Delete)
    - Routes:
      * GET /api/weatherforecast/between?from=01.01.2022&to=01.04.2022
      * GET /api/weatherforecast/week/12
      * GET /api/weatherforecast/Day?date=01.02.2022
      * POST /api/weatherforecast/Create
    - Postman? httpClient on VS code?
    - Integration test on API
    - Fill up database with fake data
    - (React application must create a lot of fake data)
* Web application
  + Create Mocks for routes by using Service worker Mock (faker)
    - As a frontend developer I need “fakeRealData”
      * Create mock route for GET /api/weatherforecast/between?from=01.01.2022&to=01.04.2022
      * Create fake data (faker.js)
      * Create mock route for GET /api/weatherforecast/week/12
      * Create fake data (faker.js)
      * Create mock route for GET /api/weatherforecast/Day?date=01.02.2022
      * Create fake data (faker.js)
      * Create mock route for POST /api/weatherforecast/Create
      * Create fake data (faker.js)
  + Search box component feature
    - As a user, I want to be able to search for cities so that I can get the weather forecast from that city.
      * Convert from city to coordinates before calling yr API
      * Fake forecast (if we don’t find other easy to use forecast provider)
      * Search component
      * Based on backend data, present city forecast
  + Show data component
    - As a user, I want to be able to favorite cites, so that they always show up in “my places”
      * Redux to persist user preferences (favorite city)
      * Show data component
    - As a user, I want the last cities I searched for to show up in “my places”, so that I don’t have to search for the same cities several times
      * Redux to persist user preferences (latest cities shown)
      * Show data component
    - As a user, I want to have a “home page” that shows my favorite cities, so that I find them quickly
      * Redux to persist user preferences (favorite city)
      * Show data component
    - As a user, I want to be able to rate the different forecast providers with a thumbs up or down, so that my preferred provider will show up on top
      * Redux to persist user preferences (rating)
      * Order the cities based on rating/alphabetical
      * On/off thumbs up button
  + Show detailed data component
    - As a user, I want to be able to click on the weather forecast for the city, so that I can get more details about the weather
      * Details page
  + Summary component
    - As a user, I want to be able to see a summary for the different forecast providers on the home page under “my places”, so that I can compare the different providers.
      * Must show comparable value in table
      * Click on city, show summary of forecast providers
      * Click on forecast provider to see more details
  + Date component
    - As a user, I want to be able to change the dates/week shown for the weather forecast, so that I can also see historic data
      * Backend: if requested from date is further back in time than data saved in DB: show oldest data instead
      * Frontend: show 10 days per page
  + Add forecast provider
    - As an admin, I want to be able to add new weather forecast providers on the website
      * Admin login
      * Form to add api info
* Azure devops
  + Create DevOps project and give access to team and owner
  + Create git (branching, how to work as a team)
  + Create pipeline for deployment
    - Web App
    - Web API
    - SQL database
    - Azure Resource (Resource group)