What Was Done

The project implemented the following features:

- Interactive Map Visualization: Displays Finnish municipalities on an interactive map that is color-coded based on migration or population data.
- Data Layer Switching: Enables users to switch between different data layers (migration and population) on the map.
- Data Retrieval from APIs: Fetches migration, population, employment and birth-death statistics from Tilastokeskus https://statfin.stat.fi/PxWeb/pxweb/en/StatFin/
- Marker Placement: Allows users to drag and drop markers on the map to select specific municipalities.
- Chart Generation: Provides charts for selected municipalities, displaying population trends, employment rates, and birth-death statistics.
- Data Combination: Enables users to combine data from two municipalities using mathematical operations and visualizes the result.
- Responsive Design: Ensures that the application is usable on both desktop and mobile devices.
- PNG Download: Allows users to download generated charts as PNG images.

Tools Used

Used HTML and CSS for web page structure and styling, JavaScript for application logic, Leaflet for interactive maps, Frappe Charts for visualizations, Leaflet-PIP to identify municipalities on the map, and the Fetch API to retrieve external data.

Points Justification

Based on the regular project requirements and the project specific features, I have listed all regular features and project specific features that are implemented in my code with the reasoning. I calculated total of 36-38 points and my points are justified as follows:

Regular features	**Points**	**Reasoning**
Well written PDF report	3 / 3	The report is clean, structured and 2 pages long
Application is responsive and can be used on both desktop and mobile environment	2 / 4 ?	Features work and scale on mobile devices, though users need to scroll in charts on small screens
Application works on Firefox, Safari, Edge and Chrome	3 / 3	Application works on listed platforms
The application has clear directory structure and everything is organized well	2 / 2	Files are organized into application.html, style.css, main.js,

		data.js, map.js, chart.js and eventHandlers.js

Project specific features	**Points**	**Reasoning**
Drag’n’drop new data to charts/maps	4 / 4	Users can drag markers onto municipalities to display and compare data.
The application show relevant data on a map and user has chance to change the data	3 / 3	Map shows migration data by default. Users can switch to population data with appropriate color hues.
The application show relevant data on a chart and user has a chance to change the data	3 / 3	Users can view chart data for each municipality and add population predictions.
User is able to switch between different layers of data on map	2 / 2?	Uncertain if migration and population data layers qualify as separate "layers."
By clicking the map user has an option to get to additional charts covering that area	4 / 4	User can get 3 charts for each municipality
There are more than one item of data available (e.g. elections data, employment rate and number of residents) – this means that there are two API calls made	3 / 3	Migration, population, employment, birth/death data
There are more than two items of data available (e.g. elections data, employment rate and number of residents) – this means that there are three API calls made	2 / 2	Migration, population, employment, birth/death data
Data is combined and merged to generate new data, which is then visualized	3 / 3	Users can place two markers on the map, generate combined population data, and visualize the results.
Users can define what should be done to different data items (e.g. values are added, multiplied together etc. before visualization)	2 / 2	Users can add, subtract, multiply, or divide combined data before visualization.
Able to download the visualization as a PNG (or SVG) image	2 / 2	Users can download a PNG image of every chart.

Declaration of AI Usage

AI assistance (OpenAI's ChatGPT, GPT-4) was used in developing this application and parts of the code. It helped in organizing and refactoring JavaScript code into modules, debugging functional issues, providing suggestions, understanding Leaflet scripts, and ensuring compliance with project requirements.