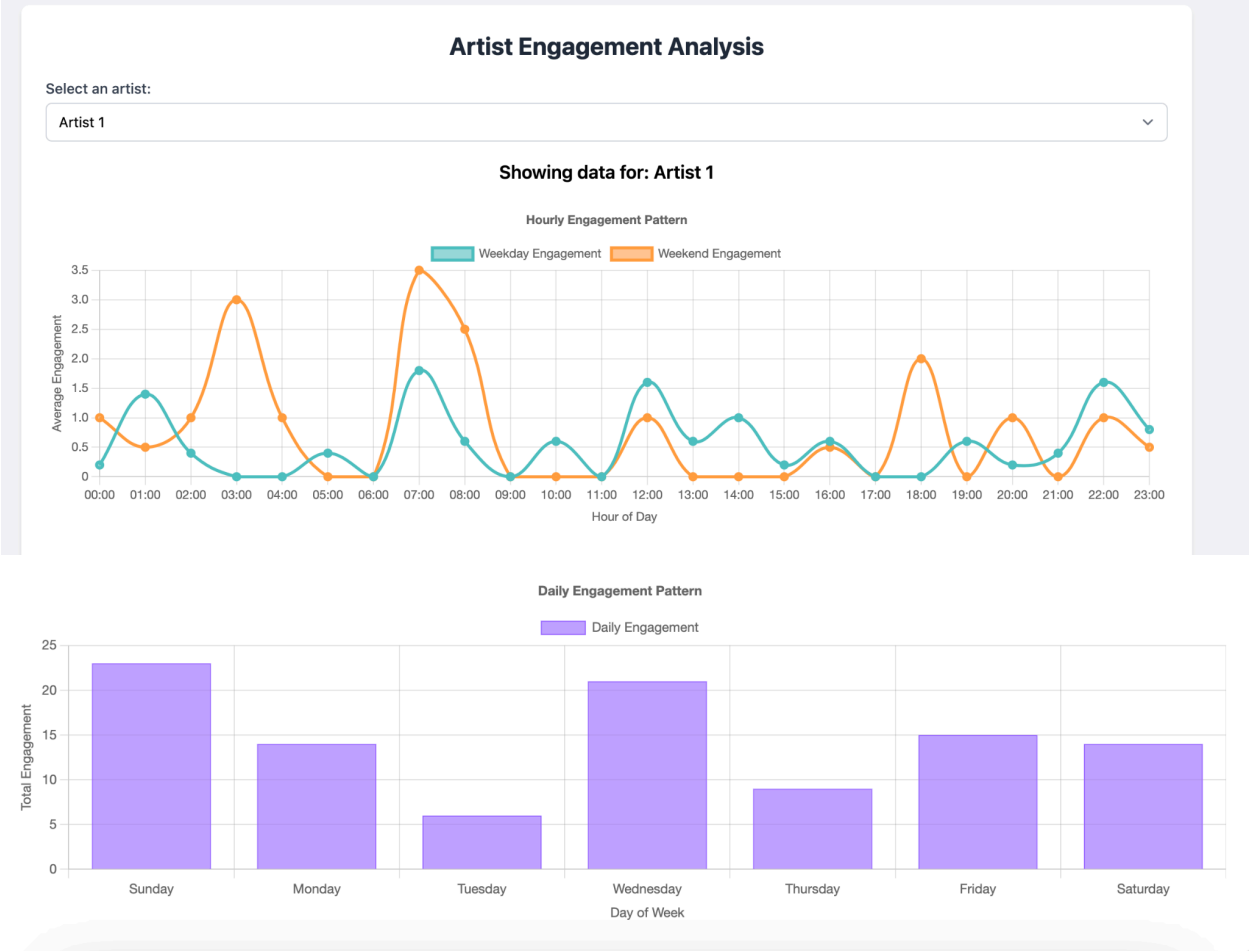
Task 1:

1. **Set Up Development Environment:**
    - Clone the repository.
    - Install required dependencies.
    - Start the development server.
2. **Analyze Database Query:**
    - Inspect the SQL query in /routes/task-1/+page.server/svelte.
    - Identify and correct errors in the query:
        - Incorrect joins
        - Incorrect calculations
        - Missing data
3. **Analyze Svelte Component:**
    - Inspect the artistTable.svelte component.
    - Identify and correct errors in the component:
        - Incorrect data binding
        - Missing calculations
        - Wrong logic
4. **Test and Debug:**
    - Run the application locally.
    - Manually test the table's data.
    - Use browser developer tools to debug issues.
    - Fix errors and re-test.

Task 2:



Data Analysis and Visualization:

1. **Data Preparation:**
    - Extract relevant data from the user_events and artist tables.
    - Choose the write data: average engagement point of every hour of a day comparing between a weekday and weekend, everyday of a week total engagement point.
    - Calculate positive engagement scores for each interaction.
2. **Data Aggregation:**
    - Group the data by artist, hour of the day, and day of the week.
    - Calculate the average positive engagement for each hour, distinguishing between weekdays and weekends but on average (weekday/5, weekend/2)
    - Calculate the total positive engagement for each day of the week.
3. **Visualization:**

    - **Line Chart:** Visualize the average engagement for each hour of the day, comparing weekdays and weekends.
    - **Bar Chart:** Visualize the total engagement for each day of the week.
4. **Frontend Implementation:**

    - Create a dropdown list to allow users to select an artist.
    - Implement two chart components to display the line chart and bar chart.
    - Fetch data from the backend API and populate the charts.
5. **Implementation**:
    - In backend area, I choose to return all the needed data at once, and pass that data to the 2 charts as props, as this would enhance performance speed and user satisfaction in small projects.

In task 2, there is a part required to consider using the user timezone. I do not think this is necessary as all timestamps are in UTC, as we do not need that function in artist page. When a user need to fetch this data in their own timezone, we can easily change this to any timezone.