The following UML class diagram illustrates the object-oriented design of the **àEmotiLog** Android application.

Class Descriptions & Responsibilities

1. Emotion

Responsibility:
Represents a predefined feeling or event that a user can log (e.g., Happy, Sad, Angry).

Description:
The class stores data about an emotion, including its name and associated icon.

Key Responsibilities:

- Store emotion metadata (name, icon)
- Act as a reusable value object for log entries

2. LogEntry

Responsibility:
Represents a single logged event created when a user presses an emotion button.

Description:
Each LogEntry records the selected Emotion along with a timestamp indicating when the log occurred. This class deals with user interaction.

Key Responsibilities:

- Store the timestamp of a log
- Associate a logged event with a specific Emotion
- Provide access to log data for summaries and storage

3. LogRepository (Interface)

Responsibility:
Defines a contract for storing and retrieving emotion logs.

Description:
LogRepository abstracts the data access layer, allowing the application to remain independent of how logs are stored.

Key Responsibilities:

- Define methods for adding log entries
- Define methods for retrieving logs by date or time period
- Separates logging logic from storage implementation

4. EmoticonLogger

Responsibility:
Implementation of the logging system.

Description:
EmoticonLogger implements the LogRepository interface and manages a collection of LogEntry objects. It handles the creation and retrieval of log data.

Key Responsibilities:

- Store and manage log entries
- Implement repository methods for querying logs
- Serve as the primary data source for summaries

5. DailySummary

Responsibility:
Generate summaries of logged emotions for a given day.

Description:
The class processes collections of LogEntry objects to calculate totals and frequency distributions of emotions for a specific date.

Key Responsibilities:

- Aggregate emotion counts for a given day
- Calculate total logs and emotion frequencies

- Present summarized data without storing raw logs

Relationships

- EmoticonLogger **implements** LogRepository to fulfill logging operations.
- EmoticonLogger **aggregates** multiple LogEntry objects.
- Each LogEntry is **associated with** exactly one Emotion.
- DailySummary **depends on** LogEntry objects to compute summaries.