mapXfit - Map your fitness!

This application is a fitness tracker that allows users to record their workouts, including type, distance, duration, and elevation gain. Users can view their workout history as a list and on a map. The app stores the workout data in local storage.

Technologies

HTML, CSS, JavaScript (Object Oriented) and Leafletjs (an open-source JavaScript library for interactive maps)

Components:

```
MapHandler: This class is responsible for handling the map functionality, including loading the map,
displaying markers for workouts, and zooming to specific locations.

LocalStorageHandler: This class manages the interaction with the local storage, storing and retrieving workout data.

UIManager: This class handles the user interface, including rendering the workout list, map markers, and popups.

FormHandler: This class manages the workout form, including handling user input and validation.

App: This class is the main entry point of the application and orchestrates the interaction between the other classes.
```

Data Flow:

```
Initialization:
    The App class creates instances of all other classes.
    The MapHandler initializes the map and registers event listeners.
    The LocalStorageHandler checks if local storage data exists.
    The UIManager initializes the UI and hides the intro section.
    The FormHandler initializes the workout form and registers event listeners.
Getting User Position:
    If there's no local storage data, the MapHandler attempts to get the user's current location using geolocation.
    Once the location is obtained, the MapHandler loads the map centered on the user's location with a specific zoom level.
Handling Local Storage Data:
    If local storage data exists, the LocalStorageHandler retrieves the data and updates the app state with the workouts.
    The UIManager renders the workout list and map markers based on the retrieved data.
Adding New Workouts:
    When the user submits the workout form, the FormHandler validates the input and creates a new workout object.
    The UIManager renders the new workout in the list and adds a marker on the map.
    The LocalStorageHandler stores the updated workout data in local storage.
User Interaction:
    Users can interact with the map by clicking on specific workout item to zoom to the workout location on the map.
    They can switch between light and dark themes using a toggle button.
```

Classes and Responsibilities:

```
MapHandler:
    Load and display the map.
    Center the map on user location or specific coordinates.
    Display markers for workouts.
    Handle map zoom and click events.
LocalStorageHandler:
    Check if local storage data exists.
    Store workout data in local storage.
    Retrieve workout data from local storage.
    Update the app state based on local storage data.
UIManager:
    Render the workout list.
    Render map markers and popups for workouts.
    Hide the intro section.
    Handle user interaction with the workout list and map.
FormHandler:
    Manage the workout form.
    Validate user input.
    Create new workout objects based on user input.
    Show and hide the form.
App:
    Orchestrate the interaction between other classes.
    Initialize all classes.
    Trigger the initial loading of the map and data depending on local storage availability.
```