Spotish

Demo time !

Link to spotish at

https://spotish.freeddns.org/

API

- `/users`
  - `POST /users` : Create a new user
  - `GET /users/{username}` : Get user info
  - `GET /users` : Get all users

API

- `/musics`
  - `GET /musics/last-listened` : Get last listened musics
  - `GET /musics/most-listened` : Get most listened musics
  - `GET /musics/{idMedia}` : Get music info by id
  - `GET /musics/{title}` : Get music info by title
  - `GET /musics` : Get all musics
  - `GET /musics/liked` : Get all liked musics from current user
  - `POST /musics/liked/{idMedia}` : Like a music for current user

API

- `/playlists`
  - `POST /playlists` : Create a new playlist
  - `GET /playlists/{idPlaylist}` : Get playlist info by id
  - `GET /playlists/user/{username}` : Get all playlists from a user
  - `GET /playlists/followed` : Get all followed playlists from current user

API

- `/playlists` (contd.)
  - `POST /playlists/followed/{idPlaylist}` : Follow a playlist for current user
  - `POST /playlists/{idPlaylist}/musics/{idMedia}` : Add a music to a playlist
  - `DELETE /playlists/{idPlaylist}/musics/{idMedia}` : Remove a music from a playlist

API

- `/creators`
  - `GET /creators/{creatorName}` : Get creator info by name
- `/albums`
  - `GET /albums/{idMedia}` : Get album info by id
- `/login`
  - `POST /login/{username}` : Login a user
  - `POST /logout` : Logout the current user

Cache - Expiration model

```
private final static int TEN_LAST_LISTENED_CACHE_MAX_AGE_SECONDS = 60;
private final static int TEN_MOST_LISTENED_CACHE_MAX_AGE_SECONDS = 600; // 10 minutes
private final static int MUSIC_ID_CACHE_MAX_AGE_SECONDS = 1800; // 30 minutes
private final static int ALL_MUSICS_CACHE_MAX_AGE_SECONDS = 3600; // 1 hour
private final static int LIKED_MUSICS_CACHE_MAX_AGE_SECONDS = 300; // 5 minutes

// Example usage for the GET /musics route
ctx.header("Cache-Control", "max-age=" + ALL_MUSICS_CACHE_MAX_AGE_SECONDS);
```

Cache - Validation model

```
private final ConcurrentMap<String, LocalDateTime> usersCache;

// Example usage for updating the cache when a playlist is created

// Store the last modification date of the user
LocalDateTime now = LocalDateTime.now();
usersCache.put(playlistCacheKey(playlistId), now);

// Invalidate the cache for all users
usersCache.remove(allPlaylistsCacheKey(ctx.cookie("userNameCookie")));
```

Cache - Validation model (contd.)

```
// Get the last known modification date of the user
LocalDateTime lastKnownModification = ctx.headerAsClass("If-Modified-Since", LocalDateTime.class).getOrDefault(null);

// Check if the user has been modified since the last known modification date
if (lastKnownModification != null && usersCache.get(playlistCacheKey(playlistId)).equals(lastKnownModification)) {
  throw new NotModifiedResponse();
}

Playlist playlist = playlistService.getPlaylist(playlistId);

LocalDateTime now;
if (usersCache.containsKey(playlistCacheKey(playlistId))) {
  // If it is already in the cache, get the last modification date
  now = usersCache.get(playlistCacheKey(playlistId));
} else {
  // Otherwise, set to the current date
  now = LocalDateTime.now();
  usersCache.put(playlistCacheKey(playlistId), now);
}

// Add the last modification date to the response
ctx.header("Last-Modified", String.valueOf(now));
```

Any questions ?