MIMUW-hats use-cases and requirements

Version 0.2, last modified 25.04.2020

Michał Borowski, Michał Makowski, Michał Tyrolski, Szymon Tworkowski

Glossary

- moderator: user assigned by system administrators whose role is removing inappropriate content
- post: image of lost item along with optional text description, user name (only with user's approval)
- feed: list of all posts containing currently lost items; users will be able to "react" to posts with a preset of Facebook-like reactions, bringing the post higher in the feed where they will be sorted according to a combination of post age and activity
- item: hat (any kind)
- valid/invalid item: item which respectively is/is not a hat

Use cases

- register account
    - Actors: future user
    - Context: A new user decides to register in our app.
    - Rationale: We require our users to be MIMUW students and/or employees to prevent content leaks, so we need to provide a secure way of registration.
    - Usage steps:
        i. Future user visits registration page, system generates a form.
        ii. The future user is asked to enter their e-mail in @(students).mimuw.edu.pl domain, system proceeds with the form only if given e-mail is valid.
        iii. Providing the e-mail was valid, a page with a nice message is displayed, asking the future user to visit given e-mail to confirm the registration by visiting a link provided there in a new message. The flow is temporarily ended (user cannot do anything from the page until he receives and visits given link).
        iv. After the future user visits a valid confirmation link from the e-mail, he is redirected to a page where a list of possible ways of authentication is given (e.g. Google, Facebook, Twitter, ~~USOS~~) with nice icons, the future user clicks on Google login icon, is redirected to Google page picks one of accounts, is redirected back to our website as a successfully registered and authenticated new user.
    - Alternative paths:
        i. In step (ii), if the given e-mail is incorrect, the future user has to reenter a valid address, staying on the same page (user receives a popup notification about invalid email) - possibly infinite loop here, until a valid e-mail is given.
        ii. After a valid e-mail is given in step (ii), but future user never confirms the received (or not) link, nothing is done to the system by far, the

future user can revisit registration page in (i) again and try to resend the link.

    - Postconditions:
        i. If the future user successfully completed all usage steps, a new account is added and his mimuw mail is reserved, so it is no longer possible to use it for registration.
        ii. Otherwise, no persistent changes are done to the system.
- register item
    - Actors: owner of item
    - Context: App's user decides to register his item in application in case of potentially losing it.
    - Rationale: Registered item, if found by other students/faculty members, can be easily associated with owner. If the owner lost their hat then someone can find it before the owner realises.
    - Preconditions:
        i. Owner of item is registered in system
    - Usage steps:
        i. Prepare file. Allowed formats: png, jpg
        ii. Uploads file to site via button
        iii. Fills optional details like (but not limited to) 'when', 'where', 'additional details'
        iv. Submits registered item.
        v. Now user can wait for system decision if image represents hat. It can take few minutes due to server overload but in most cases it should be processed immediately.
    - Alternative paths:
        i. After the last step, the system can reject the image. The user is notified and can return to the first step or request for the decision to be appealed by a moderator.
- report lost item
- log in
- report found item
- resolve lost item as found/no longer needed/etc.
- report post as inappropriate
- change notification options
- deactivate account
- prevent account from being automatically deactivated due to a period of inactivity
- prevent post from being removed due to inactivity
- view feed
- view own active posts
- view own registered items
- modify/remove own registered item
- modify/remove own post
- add/change reaction to post on feed
- moderator: verify correctness of submitted image after system has marked is as invalid and the user has appealed the decision

- moderator: respond to (manual) report of inappropriate content
- moderator: remove post
    - Actors: Moderator
    - Context: An invalid post (i. e. without hat, with xss or sql injection attack) was mistakenly approved by the system.
    - Rationale: Post harasses app or its users.
    - Preconditions: Moderator is a user with moderating permissions.
    - Usage steps:
        i. Some moderator sees this post due to users report or routine control.
        ii. Moderator clicks on ‘remove’ button to the post.
        iii. Moderator confirms action.
        iv. The system removes the post.
    - Alternative paths:
        i. After step (iv), the action can be reverted within 5 minutes.
        ii. Moderator cancels at step (ii).
- moderator: ban user

Requirements

Functional

- web frontend
- asynchronous backend
- user authentication - simple, no password
- notifying users of items being found: push notifications, email, check manually in app/website
- feed
- rewards for finding lost items (different cases: user reporting lost item gives reward to the person who found it, some reward for any valid cases of reporting found items, initial reward for registering any hat
- deactivation of accounts inactive for a set period of time; users are notified beforehand and have the option to prevent deactivation
- automatic removal of old and inactive posts
- (unimplemented) synchronization with academy cam-system
- (unimplemented) heat-map of where hats are most commonly lost in the faculty
- (unimplemented) notification/announcement of lost items by means of external bots, i.e. Twitter, Facebook, Messenger
- (unimplemented) mobile app

Non-functional

- simple, intuitive interface
- responsive web client
- gamification: incentivize users to participate by awarding points to most active ones; public rank list based on reward amount (XP/cash) and fancy MIMUW names for each level (i.e. 1 - WPI, 1707 - MD etc.)
- speed: only checking if image is valid needs to be reasonably fast (in the background, a few minutes worst-case) so the user can react quickly; other features can be slower

- language: English