Last edited on: 13. September 2021

ESE 2021 Project Requirements

Community Platform for a Common Group Interest

Students will create a community platform where people can share their stories and creations – or just repost memes on a specific topic.

Milestone 1: Learn Technologies + Implement Authentication

The first step for students is to get to know their team members and technologies. They should finish the provided implementation of the registration and login feature by modifying both the front- and backend.

Tasks for students:

- Get to know their group and technologies
  For the technologies, we provide a list of resources where they can look up the individual documentations or specific tutorials on how to implement a certain feature. We also provide two podcasts explaining the front- and backend projects of the scaffolding.
- Find a common interest within the group
  It would be good if they can decide on a common interest by the end of the first milestone once they got to know each other a bit better – it won't affect the project before the second milestone anyway.
- Finish the implementation of registration/Login of Fans (see detailed requirements below)
  We provide a working registration and login feature within the scaffolding: correct username and password will return a valid token. Two different types of users exist- default users aka fans & admins. Form validation or additional feedback will not be provided by us.
  The student's task is to extend the existing registration feature for users by adding more fields (as mentioned below) to the registration form and implement the form validation and user feedback.
  Remember that fans require registration + login, whereas admins only require login.

Detailed requirements

Admin (*already implemented by us*)

- Has access to admin-dashboard (admin dashboard is optional- can be discussed with respective assistants to understand its necessity in case the teams want to look into it) to perform upcoming admin-related features
- Admins can be created through the database (no frontend admin registration required)
- Required Fields for admin for login
  - Username OR E-Mail
  - Password

Important checks for registration:

- No duplicate Usernames OR E-Mails
- Accurate user feedback (“Username/E-mail already in use” etc.)

Fan (for all users)

The current implementation of registration has the following fields:

- Username
- Password

Students should extend this registration by adding the following fields:

- First and Last Name
- Email
- Address (incl. Street & House Number, ZIP Code, City)
- Birthday
- Phone Number

Kindly remember that you might need to extend the user model for future features (e.g. community posts, upvotes, orders, etc.)

Guest

- A user without login is a guest user

Important checks for login in addition to required checks for registration:

- Requires previous registration
- Accurate user feedback (“Username/E-Mail not found “, “A wrong Password” etc.)
- On successful login, a valid token is returned from the backend (already given in the scaffolding)

Important checks for passwords:

- Must contain capital/small letter, number, and special character
- Minimum of 8 characters
- The user gets accurate feedback which password criteria is not met
- Must be encrypted when saved into a database (not plaintext) (already given in the scaffolding)

Optional Tasks & Possible Surprise Feature ideas

- Profile Page to view/edit personal details
- Change password
- Reset password if forgotten
- Admins can CRUD other admins

Milestone 2: Community Feed/Wall Posts

The main feature of the platform is that the fans can share their stories and creations with the community. All the community posts are displayed in one place. It is up to the team how they wish to design the layout of the wall (would be discussed in the group meeting with assistants) – whether they prefer a feed similar to 9GAG/Reddit or a wall, such as Pinterest features. This milestone covers the complete implementation of CRUD (create, read, update, delete) operations (would be discussed in the group meeting with assistants) for community posts (front- and backend). Admins can update or remove

but not create community posts (would be discussed in the group meeting with assistants, simple design to keep things simple).

Detailed requirements

Students should implement the already designed model of community posts from the respective exercises.

- Posts consist of (Title + Image) OR (Title + Text) OR (Title + Image + Text)
- Title MUST be set. Text and image are optional
- One category must be set (will be discussed with assistants)
- Fans can CREATE, READ, UPDATE, DELETE their own community posts
- Admins can READ, UPDATE, DELETE any community post but cannot CREATE any
- Community posts are directly published to the feed/wall but can be removed by the admin should the content not be appropriate
- Fans can upvote or downvote other posts, admins do not.
- Each community post has a score based on its number of upvotes minus the number of downvotes (could be discussed with the assistants, students can think of their own ranking system)
- Guests can READ community posts but not up-/downvote
- Loading of posts can be infinite scrolling, pagination, or “Load More”
- Community post order should be in descending creation date or score order but can be modified by the team if they find other suitable orders.

Each post can have one or more categories, e.g. sports, Hollywood, etc.

- A predefined list of categories to set for a community post
  They can either be “built-in” statically to the site or
  OPTIONALLY dynamically be extended by admins
- Each post MUST HAVE one category assigned
  OPTIONALLY the team can decide whether fans can set multiple categories or not
- Examples for…
  - Biking Community: “Bikes”, “Accessories”, “Trips”, “Tips”…
  - Harry Potter Community: “Fan-Fiction”, “Artwork”, “Meme”…

In summary, for community Posts

- Fields to be filled out by fans
  - Title
  - Description/Text (Rich Text)
  - Image (depends on the team how they wish to implement it by looking at the scaffolding)
  - Category
- Additional Fields
  - Upvotes

    - Downvotes
    - Creation date

Kindly remember that users do not need to set anything themselves for the additional fields in the frontend. The logic will be implemented in the backend. E.g., the default timestamp will be saved when a user creates a post.

Optional Tasks & Possible Surprise Features

- User Profile Page shows their Uploads and Upvotes
- Image Upload to Server/Backend
- Fans can comment on posts
- Fans can report posts for not being appropriate
- Posts could be of different types, e.g. quiz, questions, information, etc.

We Provide

- CRUD Operations for Todo list and todo item as seen in Scaffolding
- Reading material

Milestone 3: Fan-Shop

The third milestone is to implement a simple but functioning fan shop. Students reuse their already acquired knowledge from the previous milestones and implement everything necessary on the front- and backend. A fan shop features products under different categories. For example, the fan shop of the Harry Potter Community features products under different categories, such as “Books”, “Movies”, “Soundtrack”, “Clothes”, “Figures”, “Broomsticks”, “Wands”, “Posters” and “Other”.

Use Case

- Admin receives new Nimbus 2000 replica for fan-shop: He creates new product under “Broomsticks” and enters product title, image (URL), description, and price. It is now available in the shop.
- The fan “SiriuslyNotAMuggle”, who previously logged himself in, visits the fan shop as he’s looking for the new Nimbus 2000 replica. He clicks on the “Broomsticks” category to get a selection of broomsticks and then clicks on “Nimbus 2000” to get a detailed product view with the product title/image/description/price.
- He decides to buy the product and clicks the “BUY NOW” button which opens a form to enter both the payment option and delivery address in order to complete the purchase. Under the payment, he keeps the default “Payment via invoice” option to pay the money with the bill that he receives when the product is sent to him. Under delivery address, he enters his personal address since he hasn’t previously entered it in his account. Had he entered it before, it would have been pre-filled into the form. Now that everything looks okay, he sends the form which creates a new order in the backend.
- Case 1:
    - The admin visits the “Orders” page where the new order from “SiriuslyNotAMuggle” appears. Clicking on it reveals the fan who ordered it as well as the products he ordered and the shipping address. The order status is currently set to “Pending”.

    - The admin then ships the product to the fan and changes the order status to “Shipped/Done.” (a simple admin dashboard as discussed earlier)
    - The fan “SiriuslyNotAMuggle” himself also has an “Orders” page to view all the products he previously bought and track their status. Immediately after ordering, he noticed the order status “Pending” but since the admin shipped the product, he will now see “Shipped/Done” as well.
- Case 2:
    - Immediately after sending the order, “SiriuslyNotAMuggle” decides that he should have rather bought the “Nimbus 2001” instead. He visits the “Orders” page to see his order for the “Nimbus 2000” with the status “Pending”. Since the product hasn’t been shipped yet, he is able to cancel the order which sets its status to “Cancelled”. He then later buys the “Nimbus 2001” in a new order.
    - The admin visits the “Orders” page and also sees the canceled order, but as it is canceled, he doesn’t have to do anything about it.

In summary, for Fan-Shop

- Required Product Properties
    - Title
    - Image (URL/actual image file)
    - Description
    - Price
    - Category
- The shop has a selection of appropriate categories that hold their corresponding products. Categories can either be “built-in” statically to the site or
  OPTIONALLY dynamically be extended by admins

Admin

- Only an admin can CREATE, UPDATE, DELETE store products through his admin dashboard
  Has an “Orders” page to overview all orders that have been submitted and update their state
- OPTIONALLY admins can create additional categories (similar to categories under community posts)

Fan

- Can browse the shop and its different categories for products
- Has an “Orders” page to overview all past orders that he created
- “BUY NOW” on products opens purchase form

Guest

- Can browse the shop and its different categories for products
- “BUY NOW” on products redirects to the login

Purchase/Order

- Purchase can only be triggered by a fan
- Requires fan’s chosen payment method and delivery address
    - The default payment method is “Payment via invoice”
    - Delivery address corresponds to address in the user profile
- Submitted purchase form creates order in the backend

- Order contains fan's name, payment method, delivery address, and shop product
- An order is visible to the fan who created it and all admins on their "Orders" page
- The initial order state after its creation is "Pending"
- When an order is "Pending" an admin can set it to "Shipped/Done" OR its related fan can set it to "Cancelled", both actions will terminate an order. (Ask for confirmation on these actions).

Optional Tasks & Possible Surprise Features

- Shopping cart for fans to purchase multiple items at once
- Search/Filter products (e.g. text search or filter price range)
- Implement payment services using Stripe or PayPal
- Product Review

Milestone 4: Finishing Touches

It is also time to finish any open tasks, fix bugs, and do other finishing touches.