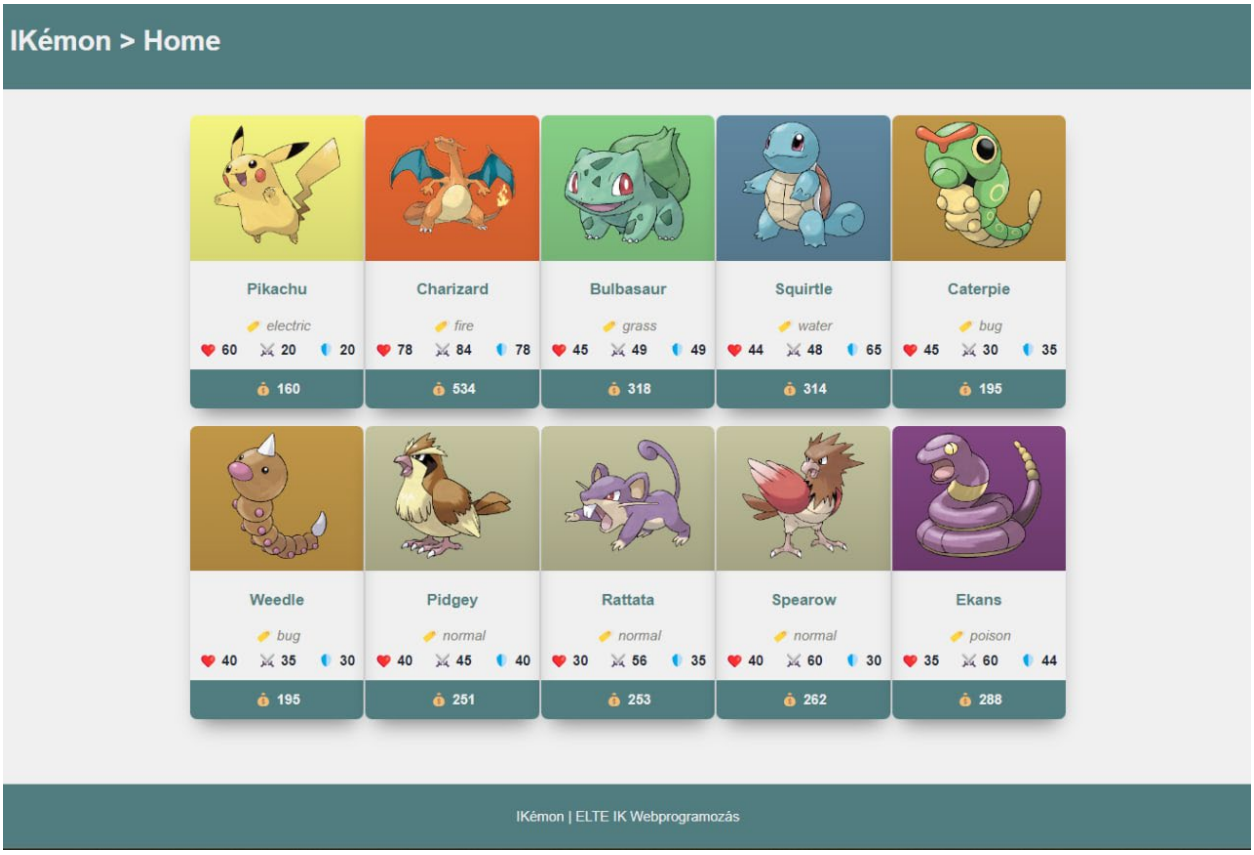
“IKémon” task



Background story

Pokémons have been released into our world! Naturally, NFT merchants quickly seized the opportunity and set about creating the most obvious way to trade, a website. They found you, and in the end they agreed that you would be responsible for building the website.

Contents

Basic functions

Data, preparations

Your task is to create a server-side application written in PHP, in which users can trade Pokémon cards. You can read about the possible storage formats for these two data structures (cards and users) in the help section below, but of course you can also structure your data in other ways.

- Pokémon cards should be prepared in the data storage in advance.
- The users should include an admin user, whose login details are fixed and who acts as a merchant. For details, see Admin functions.
- A card can only belong to one user, but a user can have more than one card.

Main page / List page

- The listing page, or main page, should display a title and a short description as static text.
- The main page is accessible to unauthenticated users who can freely browse the content displayed here.
- The list page should list the Pokémon cards that exist in the system.
- Each Pokémon card should have a link (this could be in the form of an image) to the detail page of the Pokémon card (Card details page).
- If the user is logged in, a "Buy" button should be available on cards that belong to the admin. The user can buy a card if they have enough money and if they have not yet reached a card limit (e.g. 5 cards).
- If the user is logged in, a link should be available to the "User details" page.

Card details page

- The Pokémon card details page displays the name of the monster on the card, its picture, the monster's properties, and a description of the card.
- On the page, a characteristic attribute (e.g. image background, page background color) should change according to the monster's element on the card. For example: red for Fire, yellow for Lightning, etc.
- From here you can also access the main page, and possibly other menu items.

User details page

- The page lists the user's details and his/her own cards.
- The user can sell any card for 90% of the price to the admin user. The sold card will be returned to the admin user.

Authentication pages

- The login and registration pages should be accessible from the main page.
- When registering, you must enter your username, email address and password twice. Each is mandatory, the username must be unique, the email address must be in the correct format and the passwords must match. In case of a successful registration, the user should receive X amount of money (it is recommended to burn this amount into the code, because you want all users to receive the same amount of money anyway).
- In case of registration failure, error messages should be displayed! The form should keep the state! After successful registration, the user will be logged in to the main page.
- During the login we can identify ourselves with the username and password.
- Report any errors during login above the form! After successful login, you will be redirected to the main page!

Admin functions

- Create a special user, admin (username: admin, password: admin).
- The admin user should be able to create new cards.
- The admin user can have any number of cards.
- The admin user cannot buy cards.

Extra features

See scoring section!

Other requirements

- A nice appearance is important. This doesn't necessarily mean making a page that looks fancy, but it does mean making a page that looks good at 1024x768 resolution. This can be done using a minimalist design, you can create your own CSS with different background images and graphical elements, or you can use any CSS framework.

- The input data should be checked on the server side when the request is processed. Include the novalidate attribute in the attributes of theelements of the forms to disable browser validation!
- The completed task must be uploaded to Canvas in a compressed format (zip), with all the necessary files and the README.md file in the root folder of the program, no later than the deadline.
- NO framework, external PHP library can be used to implement this. At most CSS frameworks can be used.
- The README.md file must be fully completed (see under scoring).

Help

Planning

We would also like to help those who find it more difficult to plan a larger project. It is possible to plan the whole task in advance and then develop it, but the steps below can also be used to solve smaller subtasks:

1. Create a static HTML prototype of the application to be developed. In other words, design the list page, the card detail page, etc. in the first step using only HTML and CSS. You can also try CSS statically, e.g. on the list page, how the cards should be displayed, you can burn that in statically. You can link each page together with links.
2. Think about what data you will need. What do you need to store, in what fields?
   - Where do you store users?
   - Where do you store the cards?
   - How do you store which cards belong to which users? Basically, you have two options:
     1. You store the user's purchased cards as card IDs in the user's data.
     2. You store on the card the user ID it belongs to.
3. Think about how you can extract the proper data for your pages from the data structures you have just designed. Create a function for each of these, but it is much better to implement them as methods of the Storage class.
   - How do you get back a user's cards?
   - How do you pass a card to another user?

4. For forms, there are two paths to take:
    - On success, what should happen?
    - How do I detect an error, how do I display it, how do I make a form a status holder?

Data storage

- Data: there are two types of data in the assignment: Pokémon card and User.
    - The Pokémon cards must have the following data stored:
        - Name of the Pokémon on the card
        - HP points of the Pokémon monster on the card
        - Element of the Pokémon on the card
        - Attack power of the Pokémon on the card
        - Defense of the Pokémon on the card
        - Card price
        - Description of the Pokémon on the card (optional)
        - Image of the card - this can be an image file path or image title (optional)
        - (possibly the user ID it belongs to)
    - User details:
        - User name
        - User email address
        - User's password
        - Is admin?
        - (possibly the User's cards)
- You can find all Pokémon cards are on this page, for inspiration: https://www.pokemon.com/us/pokemon-tcg/pokemon-cards/ (Links to an external site.)
- Example of card storage. It is highly recommended to use the Storage class to store the data, and not to write your own solution from scratch! This example is just a help, you have to modify it to work with that class!

```
{
  "card0": {
    "name": "Pikachu",
    "type": "electric",
```

```
        "hp": 60,
        "attack": 20,
        "defense": 20,
        "price": 160,
        "description": "Pikachu that can generate powerful electricity have cheek sacs that are extra soft and super stretchy.",
        "image": "https://assets.pokemon.com/assets/cms2/img/pokedex/full/025.png"
    },
    "card1": {
        "name": "Charizard",
        "type": "fire",
        "hp": 78,
        "attack": 84,
        "defense": 78,
        "price": 534,
        "description": "It spits fire that is hot enough to melt boulders. It may cause forest fires by blowing flames.",
        "image": "https://assets.pokemon.com/assets/cms2/img/pokedex/full/006.png"
    },
    "card2": {
        "name": "Bulbasaur",
        "type": "grass",
        "hp": 45,
        "attack": 49,
        "defense": 49,
        "price": 318,
```

```
    "description": "A strange seed was planted on its back at birth. The plant sprouts and grows with this POKéMON.",
    "image": "https://assets.pokemon.com/assets/cms2/img/pokedex/full/001.png"
  },
  "card3": {
    "name": "Squirtle",
    "type": "water",
    "hp": 44,
    "attack": 48,
    "defense": 65,
    "price": 314,
    "description": "After birth, its back swells and hardens into a shell. Powerfully sprays foam from its mouth.",
    "image": "https://assets.pokemon.com/assets/cms2/img/pokedex/full/007.png"
  },
  "card4": {
    "name": "Caterpie",
    "type": "bug",
    "hp": 45,
    "attack": 30,
    "defense": 35,
    "price": 195,
    "description": "Its short feet are tipped with suction pads that enable it to tirelessly climb slopes and walls.",
    "image": "https://assets.pokemon.com/assets/cms2/img/pokedex/full/010.png"
  },
```

```
  "card5": {
      "name": "Weedle",
      "type": "bug",
      "hp": 40,
      "attack": 35,
      "defense": 30,
      "price": 195,
      "description": "Often found in forests, eating leaves. It has a sharp venomous stinger on its head.",
      "image": "https://assets.pokemon.com/assets/cms2/img/pokedex/full/013.png"
  },
  "card6": {
      "name": "Pidgey",
      "type": "normal",
      "hp": 40,
      "attack": 45,
      "defense": 40,
      "price": 251,
      "description": "A common sight in forests and woods. It flaps its wings at ground level to kick up blinding sand.",
      "image": "https://assets.pokemon.com/assets/cms2/img/pokedex/full/016.png"
  },
  "card7": {
      "name": "Rattata",
      "type": "normal",
      "hp": 30,
```

```
        "attack": 56,
        "defense": 35,
        "price": 253,
        "description": "Bites anything when it attacks. Small and very quick, it is a common sight in many places.",
        "image": "https://assets.pokemon.com/assets/cms2/img/pokedex/full/019.png"
    },
    "card8": {
        "name": "Spearow",
        "type": "normal",
        "hp": 40,
        "attack": 60,
        "defense": 30,
        "price": 262,
        "description": "Eats bugs in grassy areas. It has to flap its short wings at high speed to stay airborne.",
        "image": "https://assets.pokemon.com/assets/cms2/img/pokedex/full/021.png"
    },
    "card9": {
        "name": "Ekans",
        "type": "poison",
        "hp": 35,
        "attack": 60,
        "defense": 44,
        "price": 288,
```

"description": "Moves silently and stealthily. Eats the eggs of birds, such as PIDGEY and SPEAROW, whole.",

"image": "https://assets.pokemon.com/assets/cms2/img/pokedex/full/023.png"

}

Scoring

You can get 20 points for solving the problem. There are minimum requirements without which the submission will not be accepted. Extra tasks can earn up to 5 additional points, but all extra tasks are worth more, so you can choose between them. In other words, if you do everything, you can get up to 30 points for the submission.

Minimum required (not accepted without them, 6 points)

- points Readme.md file: completed, uploaded
- points Main page: displayed
- points Main page: listing of all cards, e.g. with pictures
- points Main page: click on the name of the card to go to the details page of the corresponding card
- point Card details: Display the name, HP, description and element of the monster on the card
- 0.5 points Card details: The image associated with the card is displayed
- 0.5 points Card details: the colour or background colour of one or more elements on the page changes according to the element of the monster on the card, e.g. Fire is red, Lightning is yellow, etc.
- points Admin: Create new card: error handling, successful save (without authentication)

The basic tasks (14 points)

- 0.5 points Registration form: contains appropriate elements
- 0.5 points Registration form: error handling, error message, status maintenance
- 0.5 points Registration form: Successful registration
- 0.5 points Login: Error handling

- 0.5 points Login: Successful login
- 0.5 points Logout
- 0.5 points Main page: User name and money displayed
- 0.5 points Main page: Click on the username to go to the user details page
- point Main page: Allows you to filter cards by type.
- 0.5 points User details: Displays the user's name, email address, money
- 0.5 points User details: Cards associated with the user are displayed
- points User details: a sell button appears next to the user's cards, which allows the user to sell the card, the sold card is deleted from the user's cards and the user receives 90% of the price of the card. The sold card is returned to the ADMIN deck. (Where and how you place the sell button is up to you)
- 0.5 points Admin: You can log in with the admin user details
- 0.5 points Admin: New card creation is only available with Admin user
- 0.5 points Main page: When logged in, a buy button should appear under each card
- 1.5 points Main page (Buy): You can buy the card
- 0.5 points Main page (Buy): You can only buy as much as you have
- 0.5 points Main page (Buy): You can buy up to 5 cards
- point Nice design

Extra tasks (at most plus 5 points)

- 0.5 points Admin: Card modification: available to admin user for cards not yet sold
- 0.5 points Admin: Card modification: error handling, status maintenance, successful save
- point Main page: click on a button on the main page to allow non-admin users to buy a random card with their money, a random card can cost e.g.: 50 coins.
- points Main page: on the main page only 9 cards should be displayed at a time, underneath them you can navigate through the pages (with page numbers, arrows). Always display cards corresponding to the current page number, each page should display the next 9 cards. To solve this, use AJAX/fetch.
- point Exchange Step 1: On the main page, for cards that are not in the admin and are in our house, a replacement button should appear, which the user can click to replace any card with this card.

- point Exchange Step 2: The exchange should not be immediate and voluntary, but the other party should be notified and be able to accept or reject it.
- point Exchange Step 3: The exchange can include the possibility to add money to either side. Watch out for negative numbers!

README.md file

Without a properly filled README.md file, the solution will not be accepted! The README.md file must contain the following statement (<> signs are not required):

<Student's name>

<Neptun code>

Web programming - assignment

This solution was submitted and created by the student above for the Web Programming course.

I declare that this solution is my own work. I did not copy or use it from a third party

solutions from third parties. I did not forward my solution to my fellow students, nor did I publish it.

Eötvös Loránd University Student Requirements System

(Organizational and Operational Regulations of ELTE, Volume II, § 74/C) states that as long as,

as long as a student has been working on the work - or at least a significant part of it - of another student

of another student's work as his or her own, it is a disciplinary offence.

The most serious consequence of a disciplinary offence is dismissal from the university.

Minimum required (not accepted without them, 6 points)

[ ] `0.0 points` Readme.md file: completed, uploaded

[ ] `0.0 points` Main page: displayed

[ ] `1.0 points` Main page: listing of all cards, e.g. with pictures

[ ] `1.0 points` Main page: click on the name of the card to go to the details page of the corresponding card

[ ] `1.0 point` Card details: Display the name, HP, description and element of the monster on the card

[ ] `0.5 points` Card details: The image associated with the card is displayed

[ ] `0.5 points` Card details: the colour or background colour of one or more elements on the page changes according to the element of the monster on the card, e.g. Fire is red, Lightning is yellow, etc.

[ ] `2.0 points` Admin: Create new card: error handling, successful save (without authentication)

The basic tasks (14 points)

[ ] `0.5 points` Registration form: contains appropriate elements

[ ] `0.5 points` Registration form: error handling, error message, status maintenance

[ ] `0.5 points` Registration form: Successful registration

[ ] `0.5 points` Login: Error handling

[ ] `0.5 points` Login: Successful login

[ ] `0.5 points` Logout

[ ] `0.5 points` Main page: User name and money displayed

[ ] `0.5 points` Main page: Click on the username to go to the user details page

[ ] `1.0 point` Main page: Allows you to filter cards by type.

[ ] `0.5 points` User details: Displays the user's name, email address, money

[ ] `0.5 points` User details: Cards associated with the user are displayed

[ ] `2.0 points` User details: a sell button appears next to the user's cards, which allows the user to sell the card, the sold card is deleted from the user's cards and

the user receives 90% of the price of the card. The sold card is returned to the ADMIN deck. (Where and how you place the sell button is up to you)

[ ] `0.5 points` Admin: You can log in with the admin user details

[ ] `0.5 points` Admin: New card creation is only available with Admin user

[ ] `0.5 points` Main page: When logged in, a buy button should appear under each card

[ ] `1.5 points` Main page (Buy): You can buy the card

[ ] `0.5 points` Main page (Buy): You can only buy as much as you have

[ ] `0.5 points` Main page (Buy): You can buy up to 5 cards

[ ] `1.0 point` Nice design

Extra tasks (at most plus 5 points)

[ ] `0.5 points` Admin: Card modification: available to admin user for cards not yet sold

[ ] `0.5 points` Admin: Card modification: error handling, status maintenance, successful save

[ ] `1.0 point` Main page: click on a button on the main page to allow non-admin users to buy a random card with their money, a random card can cost e.g.: 50 coins.

[ ] `2.0 points` Main page: on the main page only 9 cards should be displayed at a time, underneath them you can navigate through the pages (with page numbers, arrows). Always display cards corresponding to the current page number, each page should display the next 9 cards. To solve this, use AJAX/fetch.

[ ] `1.0 point` Exchange Step 1: On the main page, for cards that are not in the admin and are in our house, a replacement button should appear, which the user can click to replace any card with this card.

[ ] `1.0 point` Exchange Step 2: The exchange should not be immediate and voluntary, but the other party should be notified and be able to accept or reject it.

[ ] `1.0 point` Exchange Step 3: The exchange can include the possibility to add money to either side. Watch out for negative numbers!