1. ***Get Banner Image (Admin)***

   *Get homepage image (// for admins)*
   *Request method: GET*
   *Request URL: '${API_BASE_URL}/mainImage/banner'*
   *Request parameter: language parameter (e.g. ?lang=zh)*
   *Example of backend response:*

```
{
    "code": 0,
    "data": [
        { "imageUrl": "http://.../zh.jpg", "id": 1 },
        { "imageUrl": "http://.../zh2.jpg", "id": 2 },
        ...
    ],
    "message": "Success"
}
```

   *For front-end display, a lang field is added for each image object when it returns.*

2. ***Add Image URL to Banner (Admin)***

   *Add a homepage image (for admins)*
   *Request method: POST*
   *Request URL: '${API_BASE_URL}/admin/addMainImages'*
   *Request parameters: { imageUrlZh, imageUrlEn }*
   *Example of backend response:*

```
{
    "code": 0,
    "data": {},
"message": "Added successfully"
}
```

3. ***Delete Image from Banner (Admin)***

   *Delete the homepage image (for admins, delete according to the image ID)*
   *Request: DELETE*
   *Request URL: '${API_BASE_URL}/admin/deleteMainImage'*
   *Request parameter: { imageId } (passed via request body)*
   *Example of backend response:*

```
{
    "code": 0,
    "image_id": {},
"message": "Deleted successfully"
}
```

4. ***Add Image File to Banner (Admin)***

   *Upload images (for admins to add images via file upload)*
   *Request method: POST*
   *URL: ${API_BASE_URL}/admin/uploadImages*

*Request parameters: FormData format, containing two files:*
*- 'image_zh': Chinese picture*
*- 'image_en': English image*

5. ***Fetch Log (Admin)***
*Get server access logs and error logs (viewed by administrators)*
*Request method: POST*
*URL: ${API_BASE_URL}/admin/log*
*Request parameters: None*
*Example of backend response:*
*{*
*"status": "success",*
*"Alogs": "2025-05-13 08:00:00 GET /index.html 200\n...",*
*"Elogs": "2025-05-13 08:00:05 ERROR Traceback...\n..."*
*}*
*@returns {Promise<{ access: string[], error: string[] } | null>}*
*Returns an object containing access logs and error logs (each item is an array of strings split by row) on success and null on failure*

6. ***Fetch Log Analyze (Admin)***
*Analyze the fields in the log file <url> and count the number of visits to each URL.*
*Return value:*
*- A JSON object where the key is the URL and the value is the number of visits to that URL.*
*- Data type: Dict[str, int]*
*- Example return value:*
*{*
*"http://example.com/page1": 5,*
*"http://example.com/page2": 3*
*}*

7. ***Get All Users (Admin)***
*Get all user information*
*Request method: GET*
*Request URL: '${API_BASE_URL}/users'*
*Request parameter: An optional language parameter, e.g. ?lang=zh*
*Example of backend response:*
*{*
*"code": 0,*
*"data": [*
*{ "id": 1, "username": "user1", "role": "user", ... },*
*{ "id": 2, "username": "admin", "role": "admin", ... }*
*],*
*"message": "query successful"*
*}*

8. ***Update User Information (Admin)***

*Update user information*

*Request method: PUT*

*Request URL: '${API_BASE_URL}/users/' // id is no longer concatenated in the URL, but is implicitly passed through the request body*

*Request parameter: language parameter (e.g. ?lang=zh)*

*Request body format (example):*

```
{
"id": 1, // passed implicitly
    "username": "newUsername",
    "role": "user"
// Other fields that need to be updated
}
```

*Example of backend response:*

```
{
    "code": 0,
    "data": { "id": 1, "username": "newUsername", "role": "user", ... },
"message": "Update successful"
}
```

9. ***Delete User (Admin)***

*Delete user information*

*Request: DELETE*

*Request URL: '${API_BASE_URL}/users/' // id is no longer concatenated in the URL*

*Request parameter: language parameter (e.g. ?lang=zh)*

*Request body format (example):*

```
{
    "id": 1
}
```

*Example of backend response:*

```
{
    "code": 0,
    "data": null,
"message": "Deleted successfully"
}
```

10. ***Create a New User (Admin)***

*Create a new user (Admin Edition)*

*Request method: POST*

*Request URL: '${API_BASE_URL}/admin/createUser'*

*Request Parameters: Language parameters (e.g. ? lang=zh)*

*Request body format (example):*

```
{
"Username": "New User",
"role": "user", // "admin" or "client"
“height”:    170,
“weight”:    60,
```

*"Age": 25,*
*"password": "newPassword"*
*}*
*Example of backend response:*
*{*
*"code": 0,*
*"data": { "id": 123, "username": "newUser", "role": "user", ... },*
*"message": "Created successfully"*
*}*

11. ***Change User's Avatar (Admin)***

*Modify user avatar (Admin interface)*
*Request method: POST*
*Request URL: '${API_BASE_URL}/admin/changeUserAvatar/${userId}'*
*Request Parameters:*
*- lang (optional, query parameter)*
*Request Body (FormData):*
*- avatar: File*

12. ***Get User's Avatar (Admin)***

*Get the avatar of the currently logged-in user*
*Request method: GET*
*Request URL: '${API_BASE_URL}/user/getAvatar'*
*Request header: Authorization: <token>Bearer*
*Return: { avatar: string }*

13. ***Get Health Suggestion (Ai)***

*Request Method: POST*

*Request URL: ${API_BASE_URL}/health-suggestion*

*Request Parameters: Language parameters (e.g. ? lang=zh)*
*Request body format (example):*

```
{
  "month": <Number>,          // Current month (1-12)
  "username": <String>,     // Username (unique identifier)
  "age": <Number>,            // User's age
  "weight": <Number>,       // User's weight in kg
  "height": <Number>,       // User's height in cm
  "lang": <String>            // Language ('en' for English, 'zh' for Chinese)
}
```

*Example Backend Response*

*Success Response (English):*

*{*
*"suggestion": "Based on your profile and the current month, I recommend drinking warm herbal teas and avoiding cold beverages. Make sure to eat more seasonal fruits and vegetables."*
*}*

*Success Response (Chinese):*

*{*
*"suggestion": "根据您的情况和当前月份，建议多喝温热的养生茶，避免冷饮，并多吃应季的水果和蔬菜。"*
*}*

14. ***Send User Message To AI (Ai)***

*Request Method: POST*

*Request URL: ${API_BASE_URL}/api/chat/BianQue*

*Request Parameters: {*
*"lang": "zh", // or "en", depending on the language preference*
*"question": "User's question here"*
*}*

*Example Backend Response: {*
*"message": "AI response to the user's question in the requested language"*
*}*

15. ***Register (Auth)***

```
/**
 * Register a new user
 * Request method: POST
 * Request URL: '${API_BASE_URL}/register'
 * Content type: application/json
 * Request body:
 * {
 *     "username": <string>,   // The username of the user
 *     "password": <string>     // The password of the user
 * }
 * Example of backend response:
 * {
 *     "msg": "Registration is successful"   // Success message
 * }
 * Error responses:
 * {
 *     "msg": "The username and password cannot be empty" // Missing fields
 * }
```

```
 * {
 *     "msg": "The username already exists" // Duplicate username
 * }
 */
```

16. ***Login (Auth)***

```
/**
 * Login a user
 * Request method: POST
 * Request URL: '${API_BASE_URL}/login'
 * Content type: application/json
 * Request body:
 * {
 *     "username": <string>,    // The username of the user
 *     "password": <string>     // The password of the user
 * }
 * Example of backend response:
 * {
 *     "access_token": <string>    // JWT token for the authenticated session
 * }
 * Error responses:
 * {
 *     "msg": "The username and password cannot be empty" // Missing fields
 * }
 * {
 *     "msg": "Wrong username or password" // Invalid credentials
 * }
 */
```

17. ***Get Profile (Auth)***

```
/**
 * Get the profile information of the logged-in user
 * Request method: GET
 * Request URL: '${API_BASE_URL}/profile'
 * Request header: Authorization: <token>Bearer
 * Example of backend response:
 * {
 *     "msg": "Welcome <username>,  This is your personal information" // Personalized welcome
message
 * }
 * Error responses:
 * {
 *     "msg": "Missing Authorization Header"   // Missing or invalid token
 * }
 */
```

18. ***Get User Information (Auth)***

```
/**
 * Get complete information about the logged-in user, including height, weight, age, etc.
 * Request method: GET
 * Request URL: '${API_BASE_URL}/user/info'
 * Request header: Authorization: <token>Bearer
 * Example of backend response:
 * {
 *     "code": 0,
 *     "data": {
 *         "id": <int>,                // User ID
 *         "username": <string>, // Username
 *         "height": <float>,       // Height in cm
 *         "weight": <float>,       // Weight in kg
 *         "age": <int>,               // Age in years
 *         "role": <string>          // Role of the user (e.g., admin, user)
 *     },
 *     "message": "Success"
 * }
 * Error responses:
 * {
 *     "msg": "Missing Authorization Header"   // Missing or invalid token
 * }
 */
```

19. ***Update User Information (Auth)***

```
/**
 * Update user information (height, weight, age)
 * Request method: PUT
 * Request URL: '${API_BASE_URL}/user/info'
 * Content type: application/json
 * Request header: Authorization: <token>Bearer
 * Request body:
 * {
 *     "height": <float>,    // Height in cm
 *     "weight": <float>,    // Weight in kg
 *     "age": <int>              // Age in years
 * }
 * Example of backend response:
 * {
 *     "code": 0,
 *     "data": {
 *         "id": <int>,                // User ID
 *         "username": <string>, // Username
 *         "height": <float>,       // Updated height
 *         "weight": <float>,       // Updated weight
```

```
*         "age": <int>                // Updated age
*     },
*     "message": "Profile updated successfully!"
* }
* Error responses:
* {
*     "msg": "Missing Authorization Header"   // Missing or invalid token
* }
*/
```

20. ***Upload Avatar (Admin)***

```
Upload a user avatar
Request method: POST
Request URL: '${API_BASE_URL}/user/avatar'
Request header: Authorization: <token>Bearer
Content type: multipart/form-data
Request body: FormData, field name avatar
Example of backend response:
{
"message": "File uploaded successfully"
}
```

21. ***Fetch Herb Area (Map)***

```
/**
 * Fetch the geographic production areas of a specific herb
 * Request method: GET
 * Request URL: '${API_BASE_URL}/api/areas/herb/<herbId>?lang=<lang>'
 * Request parameters:
 * - herbId: <int> (Required) The unique ID of the herb
 * - lang: <string> (Optional) Language preference ('zh' for Chinese, 'en' for English); defaults to 'zh'
 * Example URL:
 * - '${API_BASE_URL}/api/areas/herb/1?lang=zh'
 *
 * Example of backend response:
 * {
 *     "areas": [
 *         {
 *             "type": "Polygon",
 *             "coordinates": [
 *                 [lat1, lon1],
 *                 [lat2, lon2],
 *                 ...
 *             ]
 *         },
 *         {
 *             "type": "MultiPolygon",
 *             "coordinates": [
```

```
*             [
*                 [lat1, lon1],
*                 [lat2, lon2],
*                 ...
*             ],
*             ...
*         ]
*     }
*   ]
* }
*
* Error responses:
* - HTTP status code 500: Server error, e.g., "几何类型不支持" (Unsupported geometry type)
*/
```

22. ***Get Fuzzy Prescription (tcm)***

```
/**
* Fuzzy matching of prescriptions based on the herbs selected by the user
* Request method: POST
* Request URL: '${API_BASE_URL}/tcm/getFuzzyPrescription'
* Content type: application/json
* Request body:
* {
*   "herbs": [<int>, <int>, ...],   // Array of herb IDs selected by the user
*   "lang": <string>                // Language ('zh' for Chinese, 'en' for English); defaults to 'zh'
* }
* Example of backend response:
* {
*   "precisionResult": [
*     { "id": <int>, "name": <string> },   // Exact match prescriptions
*     ...
*   ],
*   "guessResult": [
*     { "id": <int>, "name": <string> },   // Fuzzy match prescriptions
*     ...
*   ]
* }
* Error responses:
* {
*   "error": "<string>"   // Error message if the request fails
* }
*/
```

23. ***Check Herb Selection (tcm)***

```
/**
* Check whether the herbs selected by the user meet the requirements of the specified prescription
* Request method: POST
* Request URL: '${API_BASE_URL}/tcm/checkHerbSelection'
```

```
* Content type: application/json
* Request body:
* {
*     "prescriptionId": <string>,       // ID of the prescription to check
*     "selectedHerbs": [<int>, ...], // Array of herb IDs selected by the user
*     "lang": <string>                        // Language ('zh' for Chinese, 'en' for English); defaults to 'zh'
* }
* Example of backend response:
* {
*     "result": "success" | "failure",    // Indicates whether the selection is correct
*     "message": "<string>",                    // Success or failure message
*     "lack": [<int>, ...],                    // IDs of missing herbs (if any)
*     "extra": [<int>, ...]                     // IDs of extra herbs (if any)
* }
* Error responses:
* {
*     "error": "<string>"    // Error message if the request fails
* }
*/
```

24. ***Get Herb Ids (tcm)***

```
/**
* Get an array of random herb IDs
* Request method: GET
* Request URL: '${API_BASE_URL}/tcm/herb-ids'
* Request parameters:
* {
*     "lang": <string>    // Language ('zh' for Chinese, 'en' for English); defaults to 'zh'
* }
* Example of backend response:
* {
*     "ids": [<int>, <int>, ...]    // Array of random herb IDs
* }
* Error responses:
* {
*     "error": "<string>"    // Error message if the request fails
* }
*/
```

25. ***Get Herb Detail (tcm)***

```
/**
* Get herb details based on its ID
* Request method: GET
* Request URL: '${API_BASE_URL}/tcm/herb-detail/<id>'
* Request parameters:
* {
*     "lang": <string>    // Language ('zh' for Chinese, 'en' for English); defaults to 'zh'
* }
```

```
* Example of backend response:
* {
*     "imageUrl": "<string>",    // URL of the herb image
*     "name": "<string>",          // Name of the herb
*     "options": [                    // Multiple-choice options for the quiz
*         "<string>",
*         "<string>",
*         ...
*     ]
* }
* Error responses:
* {
*     "error": "<string>"   // Error message if the request fails
* }
*/
```

26. ***Submit Quiz Score (tcm)***

```
/**
* Submit quiz test results
* Request method: POST
* Request URL: '${API_BASE_URL}/tcm/quiz-result'
* Content type: application/json
* Request body:
* {
*     "accuracy": <number>,       // Quiz accuracy percentage
*     "username": <string>,       // Username of the current user
*     "lang": <string>               // Language ('zh' for Chinese, 'en' for English); defaults to 'zh'
* }
* Example of backend response:
* {
*     "message": "Input succeed"   // Success message
* }
* Error responses:
* {
*     "error": "<string>"   // Error message if the request fails
* }
*/
```

27. ***Get All Herb Categories (tcm)***

*Description: Get all herb categories (药材分类) in the specified language.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/herbs/categories*
- *Query Parameters:*
  - *lang (string, optional, default 'zh') - language, either 'zh' or 'en'*

*Example Call:*

```
getHerbCategories(); // → ["根茎类", "叶类", "花类"]
```

*Response Example:*
*["根茎类", "叶类", "花类"]*

28. ***Get the First Twenty Herbs (tcm)***

*Description: Get the first 20 herbs (commonly used) in the specified language.*

- *Request Method: POST*
- *URL: ${API_BASE_URL}/herbs/usefulHerbs*
- *Query Parameters:*
    - *lang (string, optional, default 'zh')*

*Example Call:*
*getUsefulHerbs();*

*Response Example:*
```
[
  { "id": 1, "name": "当归", "image": "http://.../danggui.jpg" },
  { "id": 2, "name": "人参", "image": "http://.../renshen.jpg" }
  //...
]
```

29. ***Get All Herbal Flavor Classifications (tcm)***

*Description: Get all herb classifications (药材性味分类) in the specified language.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/herbs/classifications*
- *Query Parameters:*
    - *lang (string, optional, default 'zh')*

*Example Call:*
*getHerbClassifications();*

*Response Example:*
*["温性", "寒性", "甘味", "苦味"]*

30. ***Get Herbs By Category (tcm)***

*Description: Get all herbs under a certain category, in the specified language.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/herbs/${category}*
    - *Replace ${category} with the real category string, e.g. "根茎类"*
- *Query Parameters:*
    - *lang (string, optional, default 'zh')*

*Example Call:*
*getHerbsByCategory('根茎类');*

*Response Example:*

```
[
  { "id": 9, "name": "黄芪", "image": "http://.../huangqi.jpg" }
  //...
]
```

### 31. *Get Herbs By Classification (tcm)*

*Description: Get all herbs under a certain flavor/classification, in the specified language.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/herbs/classification/${classification}*
  - *Replace ${classification} with actual classification string, e.g. "温性"*
- *Query Parameters:*
  - *lang (string, optional, default 'zh')*

*Example Call:*

```
getHerbsByClassification('温性');
```

*Response Example:*

```
[
  { "id": 11, "name": "附子", "image": "http://.../fuzi.jpg" }
  //...
]
```

### 32. *Get Details of Individual Herbs (tcm)*

*Description: Get full details for a single herb by its ID, in the specified language.*

*Request Method: GET*
*URL: ${API_BASE_URL}/herbs/${id}*
*Replace ${id} with the herb ID*
*Query Parameters:*
*lang (string, optional, default 'zh')*

*Example Call:*

```
getHerbDetail(1);
```

*Response Example:*

```
{
  "id": 1,
  "name": "当归",
  "cnName": "当归",
```

```
  "category": "根茎类",
  "origin": "中国",
  "production_regions": "甘肃，四川",
  "properties": "甘温",
  "functions": "补血调经",
  "image": "http://.../danggui.jpg",
  "relate_prescription": ["归脾汤", "八珍汤"],
  "relate_prescription_id": [12, 34],
  "classification": "温性"
}
```

33. ***Search for Herbs by Name (tcm)***

*Description: Search for herbs by name (supports substring/fuzzy matching, works for Chinese & English).*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/herbs/search*
- *Query Parameters:*
  - *name (string, required): Search keyword*
  - *lang (string, optional, default 'zh')*

*Example Call:*

```
searchHerbsByName('人参');
```

*Response Example:*

```
[
  { "id": 2, "name": "人参", "image": "http://.../renshen.jpg" }
  //...
]
```

34. ***Get Herb Detail (tcm)***

*Description: Get all details of a herb for admin use, including both Chinese and English.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/admin/getHerbDetails/${id}*
- *Query Parameters:*
  - *lang (string, optional, default 'zh')*

*Example Call:*

```
getHerbDetailAdmin(1)
```

*Response Example:*

```
{
  "id": 1,
  "name_en": "Angelica sinensis",
  "name_zh": "当归",
  "category_en": "Root",
  "origin_en": "China",
```

```
  "production_regions_en": "Gansu, Sichuan",
  "properties_en": "Sweet, Warm",
  "functions_en": "Blood tonic, regulate menstruation",
  "image_en": "http://.../danggui_en.jpg",
  "classification_en": "Warm",
  "category_zh": "根茎类",
  "origin_zh": "中国",
  "production_regions_zh": "甘肃，四川",
  "properties_zh": "甘温",
  "functions_zh": "补血调经",
  "image_zh": "http://.../danggui.jpg",
  "classification_zh": "温性"
}
```

### 35. *Get All Prescriptions (tcm)*

*Description: Retrieve a list of all prescriptions (ID, name only) in the specified language.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/prescriptions*
- *Query Parameters:*
  - *lang (string, optional, default: 'zh') - 'zh'/Chinese or 'en'/English*

*Frontend Call Example:*

```
getPrescriptions('zh'); // or 'en'
```

*Sample Response:*

```
[
  { "id": 1, "name": "四君子汤" },
  { "id": 2, "name": "六味地黄丸" }
]
```

### 36. *Get Individual Prescription Details (tcm)*

*Description: Get full details of a single prescription by ID, in the specified language.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/prescriptions/${id}*
  - *Replace ${id} with the prescription's ID.*
- *Query Parameters:*
  - *lang (string, optional, default: 'zh')*

*Frontend Call Example:*

```
getPrescriptionDetail(2, 'zh');
```

*Sample Response:*

```
{
  "id": 2,
```

```
  "name": "六味地黄丸",
  "cnName": "六味地黄丸",
  "constitute": "熟地黄; 山药; 山茱萸; 泽泻; 牡丹皮; 茯苓",
  "action": "补肾滋阴",
  "indication": "肾阴虚证",
  "constituteId": [5, 6, 7, 8, 9, 10]
}
```

### 37. *Admin Get Prescription Details (tcm)*

*Description: As an admin, get all information for a prescription in both English and Chinese.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/admin/getPrescriptionDetails/${id}*
- *Query Parameters:*
  - *lang (string, optional, default: 'zh')*

*Frontend Call Example:*

```
getPrescriptionDetailAdmin(2, 'zh')
```

*Sample Response:*

```
{
  "id": 2,
  "name_en": "Liuwei Dihuang Pill",
  "name_zh": "六味地黄丸",
  "constitute_en": "Rehmannia; Dioscorea; Cornus; Alisma; Moutan; Poria",
  "action_en": "Tonify kidney Yin",
  "indication_en": "Kidney Yin deficiency",
  "constitute_zh": "熟地黄; 山药; 山茱萸; 泽泻; 牡丹皮; 茯苓",
  "action_zh": "补肾滋阴",
  "indication_zh": "肾阴虚证"
}
```

### 38. *Search Prescriptions by Name (tcm)*

*Description: Search for prescriptions with a name match (supports fuzzy/substring search, works for both Chinese and English).*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/prescriptions/search*
- *Query Parameters:*
  - *name (string, required): The search keyword*
  - *lang (string, optional): 'zh' or 'en'*

*Frontend Call Example:*

```
searchPrescriptionsByName('君子', 'zh')
```

*Sample Response:*

*[*
*{ "id": 1, "name": "四君子汤" }*
*]*

*Error Example (No Input):*
*{ "error": "No name provided" }*

### 39. *Admin: Add a Prescription (tcm)*

*Description: Add a new prescription (admin operation). Supports both English and Chinese prescription data.*

- *Request Method: POST*
- *URL: ${API_BASE_URL}/admin/addPrescription*
- *Query Parameters:*
  - *lang (string, optional): 'zh' or 'en'*
- *Body (JSON):*
  - *All of: name_en, name_zh, constitute_en, constitute_zh, action_en, action_zh, indication_en, indication_zh*
  - *action: 1 (for adding, see update below)*

*Frontend Call Example:*

```
addPrescription({
  name_en: "Liuwei Dihuang Pill",
  name_zh: "六味地黄丸",
  constitute_en: "Rehmannia; Dioscorea; Cornus; Alisma; Moutan; Poria",
  constitute_zh: "熟地黄；山药；山茱萸；泽泻；牡丹皮；茯苓",
  action_en: "...",
  action_zh: "...",
  indication_en: "...",
  indication_zh: "...",
  action: 1 // "add"
});
```

*Sample Response:*
*{ "message": "Success!" }*

### 40. *Admin: Update a Prescription (tcm)*

*Description: Update an existing prescription (admin operation).*

- *Request Method: POST*
- *URL: ${API_BASE_URL}/admin/addPrescription*
- *Query Parameters:*
  - *lang (string, optional)*
- *Body (JSON):*
  - *All fields as above, must include id*

- *action: 0 (for update)*

*Frontend Call Example:*

```
updatePrescription({
  id: 2,
  name_en: "...",
  name_zh: "...",
  ...,
  action: 0 // "update"
});
```

*Sample Response:*

```
{ "message": "Success!" }
```

### 41. *Admin: Delete a Prescription (tcm)*

*Description: Delete a prescription (admin operation), removes both English and Chinese versions and cleans up associations.*

- *Request Method: DELETE*
- *URL: ${API_BASE_URL}/admin/deletePrescription*
- *Body (JSON):*
  - *id (number): Required, prescription id to delete*
  - *lang (string, optional)*

*Frontend Call Example:*

```
deletePrescription(2, 'zh')
```

*Sample Response:*

```
{ "message": "Prescription group and related herb links deleted successfully!" }
```

### 42. *Get a Prescription Quiz Question (tcm)*

*Description: Fetch a prescription question for the quiz, based on language and difficulty, avoiding IDs already presented.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/tcm/prescriptionQuiz/question*
- *Query Parameters:*
  - *lang (string, required): Language, e.g. zh or en*
  - *difficulty (number/string, required): Difficulty level, 1 (easy), 2 (medium), 3 (hard)*
  - *doneQuestionIds (string, optional): Comma-separated list of completed question IDs, e.g. '5,17,29'*

*Frontend Call Example:*

```
getPrescriptionQuestion('zh', 2, [5, 17, 29]);
```

*Sample Response:*

*{*
*"id": 12,*
*"prescriptionName": "六味地黄丸",*
*"correctHerbIds": ["熟地黄", "山药", "山茱萸", "泽泻", "牡丹皮", "茯苓"],*
*"difficulty": "2"*
*}*

### 43. *Submit a Prescription Quiz Score (tcm)*

*Description: Submit a score for a prescription quiz session.*

- *Request Method: POST*
- *URL: ${API_BASE_URL}/tcm/score*
- *Request Body (JSON):*
  - *accuracy (number, required): The accuracy (%) from the quiz*
  - *username (string, required): The current username ("", or actual username)*
  - *totalTime (number, required): Total time spent (seconds)*
  - *lang (string, required): Language, e.g. zh or en*

*Frontend Call Example:*

```
submitPrescriptionQuizScore(83, 'johndoe', 95, 'en');
```

*Sample Response:*
*{ "message": "Input succeed" }*
*Error Example (no user):*
*{ "message": "No available user" }*

### 44. *Get an Array of Herb IDs (tcm)*

*Description: Fetch 5 random herb IDs for quiz purposes.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/tcm/herb-ids*
- *Query Parameters:*
  - *lang (string, optional, default: zh)*

*Frontend Call Example:*

```
getHerbIds('zh');
```

*Sample Response:*
*{ "ids": [1, 27, 5, 99, 140] }*

### 45. *Get Herb Details for Quiz (tcm)*

*Description: Fetch image and four names (one correct) for a given herb ID—used in "identify the herb" quiz.*

- *Request Method: GET*
- *URL: ${API_BASE_URL}/tcm/herb-detail/${id}*

- *Replace ${id} with the desired herb ID.*

- *Query Parameters:*
  - *lang (string, optional, default: zh)*

*Frontend Call Example:*

```
getHerbDetail(27, 'zh');
```

*Sample Response:*

```
{
  "imageUrl": "/static/27.jpg",
  "name": "当归",
  "options": ["黄芪", "白芍", "当归", "川芎"]
}
```

### 46. *Submit Quiz Test Results (tcm)*

*Description: Submit "herb recognition" quiz results, including accuracy, username, and time spent.*

- *Request Method: POST*
- *URL: ${API_BASE_URL}/tcm/quiz-result*
- *Request Body (JSON):*
  - *accuracy (number, required): The accuracy (%) from the quiz*
  - *username (string, required): User's name*
  - *totalTime (number, required): Total time spent (seconds)*
  - *lang (string, optional, default zh)*

*Frontend Call Example:*

```
submitQuizScore(95, 'johndoe', 60, 'en');
```

*Sample Response:*

```
{ "message": "Input succeed" }
```

*Error Example (no user):*

```
{ "message": "No available user" }
```

### 47. *Validate Herb Selection for a Prescription (tcm)*

*Description: Submit user's selection of herb IDs for a prescription and receive feedback on correctness, lacking, or extra herbs.*

- *Request Method: POST*
- *URL: ${API_BASE_URL}/tcm/checkHerbSelection*
- *Request Body (JSON):*
  - *prescriptionId (number, required)*
  - *selectedHerbs (array of numbers, required)*
  - *lang (string, required)*

*Sample Response: (Correct)*

```
{
```

```
    "result": "success",
    "message": "恭喜你，选择正确",
    "lack": [],
    "extra": []
}
```


*Sample Response: (Lacking herbs)*


```
{
    "result": "failure",
    "message": "缺少药材{'熟地黄'}",
    "lack": [12],
    "extra": []
}
```


*Sample Response: (Extra herbs)*


```
{
    "result": "failure",
    "message": "药材{'黄芪'}是多余的",
    "lack": [],
    "extra": [27]
}
```


### 48. *Medicinal Herb Image Test (QUIZ) Ranking List (tcm)*

*Description:*
*Fetch the top 10 user rankings for the "herb identification/quiz" module. Rankings are sorted by a combined score based on accuracy and time.*

- *Request Method: GET*
- *URL: /api/tcm/ranking*
- *Query Parameters: None*

*Example Usage:*

```
const data = await fetchRankingData();
```

*Example Response:*


```
{
    "status": "success",
    "data": [
      {
          "username": "alice",
          "avatar": "/static/avatar1.jpg",
          "accuracy": 92,
          "score": 85
      },
      {
          "username": "bob",
```

```
        "avatar": "/static/avatar2.jpg",
        "accuracy": 90,
        "score": 83
      }
      // ... up to 10 entries
    ]
}
```

*Field Descriptions:*

- *username (string): The player's username*
- *avatar (string): User's avatar image path*
- *accuracy (number): Average answer accuracy (%)*
- *score (number): Calculated as: Math.max(0, accuracy - 0.03 * totalTime) (integer, non-negative)*

### 49. *Prescription Test (Quiz) Leaderboard (tcm)*

*Description:*
*Fetch the top 10 user rankings for the "prescription quiz" module. Sort is also based on combined accuracy and time.*

- *Request Method: GET*
- *URL: /api/tcm/ranking/prescription*
- *Query Parameters: None*

*Example Usage:*

```
const data = await fetchPrescriptionRankingData();
```

*Example Response:*

```
{
  "status": "success",
  "data": [
    {
      "username": "carol",
      "avatar": "/static/avatar3.jpg",
      "accuracy": 95,
      "score": 89
    },
    {
      "username": "dave",
      "avatar": "/static/avatar4.jpg",
      "accuracy": 88,
      "score": 80
    }
    // ... up to 10 entries
  ]
}
```

*Field Descriptions:*

- *username (string): The player's username*
- *avatar (string): User's avatar image path*
- *accuracy (number): Average answer accuracy (%)*
- *score (number): Calculated as: Math.max(0, accuracy - 0.03 * totalTime) (integer, non-negative)*

### 50. *Fetch Story Data (tcm)*

*Description:*
*Retrieve the full data of a specific story (episode/scene) node.*

- *Request Method: GET*
- *URL: /api/story/{storyId}*
    - *Replace {storyId} with the numeric ID of the story node you want to fetch.*

*Example Usage:*

```
const data = await fetchStoryData(1);
```

*Example Response:*

```
{
  "story_number": 1,
  "id": 1,
  "label": "药铺",
  "speaker": "医师",
  "dialog": "这里是故事内容",
  "choices": ["购买药材", "离开"],
  "next_id": [2, 3],
  "background": "/static/bg1.jpg",
  "character_image": ["/static/doctor.png"],
  "audio": "/static/line1.mp3",
  "speaker_en": "Doctor",
  "dialog_en": "This is the story text.",
  "choices_en": ["Buy herbs", "Leave"]
}
```

*Error Response:*

```
{ "message": "Story not found" }
```

### 51. *Fetch Next Story Data (tcm)*

*Description:*
*Given the current scene/story node ID, retrieve an array of all possible "next" story nodes that the player can choose from.*

- *Request Method: GET*
- *URL: /api/story/next/{storyId}*
    - *Replace {storyId} with the current story node ID.*

*Example Usage:*
*const nextStories = await fetchNextStoryData(1);*

*Example Response:*
*[*
*{*
*"story_number": 2,*
*"id": 2,*
*"label": "市场",*
*"speaker": "商贩",*
*"dialog": "你来到市场。",*
*"choices": ["还价", "购买"],*
*"next_id": [4, 5],*
*"background": "/static/bg2.jpg",*
*"character_image": ["/static/vendor.png"],*
*"audio": "/static/line2.mp3",*
*"speaker_en": "Vendor",*
*"dialog_en": "You arrive at the market.",*
*"choices_en": ["Bargain", "Buy"]*
*},*
*{*
*"story_number": 3,*
*"id": 3,*
*"label": "药铺外",*
*"speaker": "同伴",*
*"dialog": "你离开了药铺。",*
*"choices": ["返回药铺", "去巷子"],*
*"next_id": [1, 8],*
*"background": "/static/bg3.jpg",*
*"character_image": ["/static/friend.png"],*
*"audio": "/static/line3.mp3",*
*"speaker_en": "Companion",*
*"dialog_en": "You left the pharmacy.",*
*"choices_en": ["Return to pharmacy", "Go to alley"]*
*}*
*]*

*Error Response:*
*{ "message": "Story not found" }*

52. ***Get Herb names by Herb Ids***

*/**
* Batch Get Herb Names by IDs
*
* Request method: POST
* Request URL: `${API_BASE_URL}/herbs/batchNames`
* Request body:*

```
 *   {
 *       ids: [1, 2, 3],
 *       lang: "zh"
 *   }
 *
 * Response example:
 *   [
 *       { id: 1, name: "黄芪" },
 *       { id: 2, name: "当归" },
 *       { id: 3, name: "白术" }
 *   ]
 *
 * Errors:
 *   返回 null 并打印错误信息
 */
export const getHerbNamesBatch = async (ids, lang = 'zh') => {
    try {
        const response = await axios.post(`${API_BASE_URL}/herbs/batchNames`, {
            ids,
            lang
        });
        return response.data;   // 假设返回直接是数组
    } catch (error) {
        console.error(`批量获取药材名称失败: ${error.message}`);
        return null;
    }
};
```