Task for TypeScript (JavaScript also) developer. Develop an application

Step 1

Write a program with TypeScript (version 4.7 and newer), that meets the following criteria:

- REST Api
- 2. Response JSON models (headers: Content-Type: application/json; charset=UTF-8)
- 3. Write main unit tests for each component
- 4. Must contains selected components (the list is below)
- 5. Server for responses can be Mock on Postman
- 6. Application must be compiled from TypeScript to JavaScript with Laravel Mix or ViteJS (NodeJs packages)

Application is a web client to simulate requests below. Application must be OOP. All arguments/parameters and variables must be have type annotations. All models (below) must have models \${Model}.d.ts files. For UI you can make simple html files (also with simple css styles) (without usage of external libraries like bootstrap)

list of components

- 1. User registration
- 2. Posts
- 3. Comments

Components

Registration

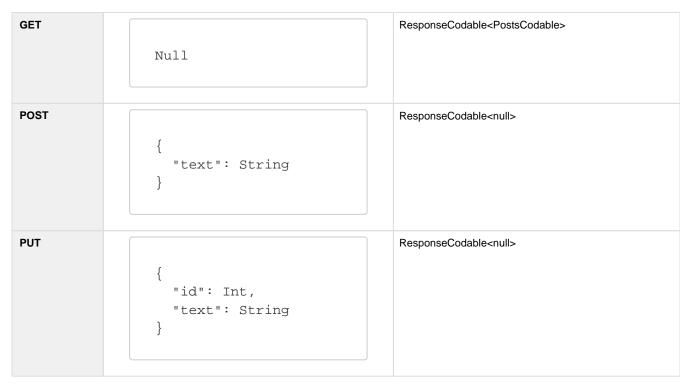
url: /api/v1.0/account

Method	Input	Output
GET	Null	ResponseCodable <accountcodable></accountcodable>
POST	<pre>{ "username": String, "password": String }</pre>	ResponseCodable <null></null>
PUT	{ "password": String }	ResponseCodable <null></null>

Posts

url: /api/v1.0/posts

Method	Input	Output
--------	-------	--------



Comments

url: /api/v1.0/posts/{id}/comments

Method	Input	Output
GET	Null	ResponseCodable <commentscodable></commentscodable>
POST	{ "text": String }	ResponseCodable <null></null>
PUT	<pre>{ "id": Int, "text": String }</pre>	ResponseCodable <null></null>

Models

✓ Model ResponseCodable<T>

```
struct ResponseCodable<T>: Codable {
  var data: T?
  var errors: [ErrorCodable]?
  // var status: Int // HTTP Response Code
}
```

→ Model ErrorCodable

```
struct ErrorCodable: Codable {
  var code: String
  var message: String?
}
```

Model AccountCodable

```
struct AccountCodable: Codable {
  var id: Int
  var username: String?
}
```

Model PostsCodable

```
struct PostsCodable: Codable {
  var posts: [PostCodable]?
}
```

Model PostCodable

```
struct PostCodable: Codable {
  var id: Int
  var owner: Int? // User.id (publisher)
  var text: String?
  var created_at: String? // (datetime)
  var updated_at: String? // (datetime)
}
```

Model CommentsCodable

```
struct CommentsCodable: Codable {
  var comments: [CommentCodable]?
}
```

→ Model CommentCodable

```
struct CommentCodable: Codable {
 var id: Int
  var owner: Int? // User.id (publisher)
 var text: String?
 var created_at: String? // (datetime)
 var updated_at: String? // (datetime)
 var module: String? // posts
 var module_id: Int? // post.id
```

Step 2

Source:

Object: {"list": [{"id": 1, "name": "First"}, {"id": 2, "name": "Second"}]}

Models:

```
type ListModel = {
 list: ListItemModel[]
type ListItemModel = {
 id: number
 name?: string | null
```

Task:

Make a class For Each, that will foreach object.list with automatic annotations and types strictions from source variable, ex: object.list



```
class SomeClass {
  private object: ListModel = {"list": [{"id": 1, "name": "First"}, {"id": 2, "name": "Second"}]}
  private object2: ListModel = {"list": [{"id": 3, "name": "First"},
  {"id": 4, "name": "Second"}]}

  constructor() {
    ForEach(object.list ?? [], 'id', item => {
        console.debug(item)
    })

    ForEach(object2.list ?? [], 'name', (item, key) => {
        console.debug(item, key)
    })
  }
}
```

Wrong

```
class SomeClass {
  private object: ListModel = {"list": [{"id": 1, "name": "First"}, {"id": 2, "name": "Second"}]}

  constructor() {
    ForEach(object.list ?? [], 'id', item => { // Good console.debug(item) })

    ForEach(object.list ?? [], 'title', item => { // Exception, variable title is not in array rows console.debug(item) })
    })
}
```