Documentazione valutazione docenti

Alba Luca Francesco

Aprile 2024

Indice:

1 Dettagli Frontend

1.1 Accesso

1.1.1 '/login'

Riceve come parametri in POST:

- user
- password

In output:

```
{
    credenziali_res: true/false,
    tuo_token: tk/undefined,
    messaggio: 'Benvenuto su valutazione docenti!'
}
```

Se tutto è valido, salvare il token nel `sessionstorage`, `localstorage`, o in un cookie. Inoltre, è consigliabile salvare sempre nel `sessionstorage`, `localstorage`, o in un cookie un flag che viene impostato su true e false se l'utente è loggato oppure no.

1.1.2 Controllo ruolo dell'utente

Per controllare il ruolo dell'utente che ha effettuato l'accesso, utilizzare l'endpoint `'/ruolo_utente'` che riceve come parametri in POST:

- token

In output:

```
{
    tipo: 'A'/'S'/'D'/'',
    messaggio: 'Ruolo trovato con successo'
}
```

1.1.3 Controllo prima di chiamare qualsiasi endpoint

Prima di richiamare qualsiasi endpoint del server, è necessario richiamare sempre l'endpoint `'/token_valido'`, il quale controlla se il token è ancora valido o meno, si consiglia di creare una funzione il quale restituirà una promise per facilitare i relativi controlli successivi. Clicca qui per vedere un esempio, guarda dalla riga 30 alla riga 44.

1.1.4 '/logout'

Riceve come parametri in POST:

- token

In output:

```
{
    successo: false,
    messaggio: "Autenticazione fallita"
}
```

Questo endpoint serve a sloggare l'utente dal suo account, quindi rimuovere il token dal `sessionstorage`, `localstorage`, o dal cookie.

1.2 Studente

1.2.1 '/get_docenti_classe'

Riceve come parametri in POST:

- token

```
In output:
{
    docenti: docente/null, → [{nome: ‘’, cognome: ‘’, materie: mat}, ...]
    messaggio: "I dati del docente sono stati estratti con successo!",
    valuta: true/false/null
}
```

Questo endpoint serve a caricare i docenti dell'alunno in questione. Clicca qui per vedere un esempio, guarda dalla riga 179 alla riga 212.

1.2.2 '/get_domande'

Non riceve parametri in GET. In output restituirà:

```
{
    domande: array_domande/null,
    messaggio: 'Domande estratte!'
}
```

Questo endpoint serve a ricevere le domande da porre allo studente. Clicca qui per vedere un esempio, guarda dalla riga 103 alla riga 116.

1.2.3 '/valuta_docente'

Riceve come parametri in POST:

- nome_docente
- cognome_docente
- token
- valutazioni → [{idDomanda: '0001', voto: 10}, ...]

```
In output:
{
    messaggio: "Valutazione inviata!"
}
```

Questo endpoint serve a inviare le valutazioni dello studente. Clicca qui per vedere un esempio, guarda dalla riga 118 alla riga 147.

1.3 Admin

1.3.1 '/view_docente'

Riceve come parametri in POST:

- nome_docente
- cognome_docente
- token

```
In output:
{
    media: media_voti/null, → [{domanda: '001', media: '7.5'}, ...]
    messaggio: "Non sei autorizzato ad accedere a questa risorsa!"
}
```

Questo endpoint serve a ricevere le medie di un docente specifico. Clicca qui per vedere un esempio, guarda dalla riga 254 alla riga 286.

1.3.2 '/get_docenti'

Riceve come parametri in POST:

- token

```
In output:
{
    docenti: risultato/null, → [{nome, cognome, email}, ...]
    messaggio: "Dati dei docenti ottenuti con successo!",
}
```

Questo endpoint serve a caricare il nome ed il cognome dei docenti in una select. Clicca qui per vedere un esempio, guarda dalla riga 288 alla riga 319.

1.3.3 '/carica_docenti'

Riceve come parametri in POST:

- token

```
In output:
{
    messaggio: "Docenti caricati con successo"
}
```

Questo endpoint serve a caricare i docenti nel db. Clicca qui per vedere un esempio, guarda dalla riga 344 alla riga 365.

1.3.4 '/carica_studenti'

Riceve come parametri in POST:

- token

```
In output:
{
    messaggio: "Studenti caricati con successo"
}
```

Questo endpoint serve a caricare gli studenti nel db.

1.3.5 '/admin_console'

Riceve come parametri in POST:

- token

```
In output:
[
    {
        button: "Carica Studenti",
        urlEndpoint:"carica_studenti"
    },
    {
        button: "Carica Docenti",
        urlEndpoint:"carica_docenti"
    },
    {
        button: "Start/Stop Valutazioni",
        urlEndpoint: "start_stop_valutazioni"
    },
    {
        status: valutazioni_avviate
    }
]
```

Questo endpoint serve a caricare la dashboard dell'admin. Clicca qui per vedere un esempio, guarda dalla riga 367 alla riga 410.

1.3.6 '/start_stop_valutazioni'

Riceve come parametri in POST:

- token

```
In output:
{
    valuta: true/false/null,
    messaggio: "Valutazioni avviate"
}
```

Questo endpoint serve a iniziare o terminare le valutazioni. Clicca qui per vedere un esempio, guarda dalla riga 412 alla riga 439.

1.4 Docente

1.4.1 '/get_nome_cognome_docente'

Riceve come parametri in POST:

- token

```
In output:
{
    nome: 'Manuela',
    cognome: 'Dalbesio',
    messaggio: 'Nome e cognome trovati!'
}
```

Questo endpoint serve a ricevere il nome ed il cognome del docente che ha fatto accesso alla piattaforma. Clicca qui per vedere un esempio, guarda dalla riga 445 alla riga 495.

1.4.2 '/ scarica_pdf_valutazioni '

Riceve come parametri in POST:

- nome_docente
- cognome_docente
- valutazioni
- token

In output riceve un blob.

Questo endpoint serve a scaricare il PDF per le medie del docente. Clicca qui per vedere un esempio, guarda dalla riga 497 alla riga 541.