REST API – RESERVES

1. Model Espais

a) Action List

⇒ Request

- URL: http://localhost/yii/reserves/api/espais
- Filters: ?codi=A&descripcio=Espai%
- Method: GET
- Headers: X_USERNAME: username
  X_PASSWORD: password_encriptat

⇒ Response

- Return: Devuelve los espacios que puede ver el usuario. Todos, si es usuario admin.
- JSON:

  [{"id": "1", "codi": "A", "descripcio": "Espai A"},
  {"id": "2", "codi": "B", "descripcio": "Espai B"}]

b) Action View

⇒ Request

- URL: http://localhost/yii/reserves/api/espais/{id}
- Method: GET
- Headers: X_USERNAME: username
  X_PASSWORD: password_encriptat

⇒ Response

- Return: Devuelve el espacio correspondiente al id indicado en la URL. Si el espacio no existe o el usuario no lo puede ver, devuelve error 404.
- JSON:

{"id": "1", "codi": "A", "descripcio": "Espai A"}

- Errors:

{"codError": 404, "descError": "VIEW: No existeix o no es pot accedir a l'espai amb id=100"}

c) Action Create

⇒ Request

- URL: http://localhost/yii/reserves/api/espais/
- Method: POST
- Headers: Content-Type: application/x-www-form-urlencoded
  X_USERNAME: username
  X_PASSWORD: password_encriptat
- Data: codi=XX&descripcio=Descripció+XX

⇒ Response

- Return: Devuelve id del espacio creado. Si el usuario no es admin, devuelve error 403.
- JSON:

  {"id": "11"}
- Errors:

  {"codError": 403, "descError": "CREATE: L'usuari no té permís per crear espais"}

d) Action Update

⇒ Request

- URL: http://localhost/yii/reserves/api/espais/{id}
- Method: PUT
- Headers: Content-Type: application/json

X_USERNAME: username

X_PASSWORD: password_encriptat

Data: {descripcio: “Espai XX”}, etc.

o bien {usuari_id: “2”} para dar permiso espacio-usuario

⇒ Response

- Return: Devuelve id del espacio actualizado. Si el usuario no es admin, devuelve error 403. Si no existe el espacio, devuelve error 404.
- JSON:

  {"id": "11"}
- Errors:

  {"codError": 403, "descError: "UPDATE: L'usuari no té permís per actualitzar espais"}

  {"codError": 404, "descError": "UPDATE: No existeix l'espai amb id=100"}

e) Action Delete

⇒ Request

- URL: http://localhost/yii/reserves/api/espais/{id}
- Method: DELETE
- Headers: X_USERNAME: username

  X_PASSWORD: password_encriptat

⇒ Response

- Return: Devuelve id del espacio borrado. Si el usuario no es admin, devuelve error 403. Si no existe el espacio, devuelve error 404.
- JSON:

  {"id": "11"}
- Errors:

  {"codError": 403, "descError: "DELETE: L'usuari no té permís per esborrar espais"}

  {"codError": 404, "descError": "DELETE: No existeix l'espai amb id=100"}

2. Model Usuaris

a) Action List

⇒ Request

- URL: http://localhost/yii/reserves/api/usuaris
- Filters:

  ?nom_usuari=user%&nom=nom%&cognoms=cognoms%&es_admin=1
- Method: GET
- Headers: X_USERNAME: username

  X_PASSWORD: password_encriptat

⇒ Response

- Return: Devuelve los usuarios que puede ver el usuario. Todos, si es usuario admin. Si no es admin, devuelve el propio usuario.
- JSON:

  [{"id": "1", "nom_usuari": "user", "contrasenya": "password", "nom": "nom", "cognoms": "cognoms", "es_admin": "1"},

  {"id": "2", "nom_usuari": "user2", "contrasenya": "password2", "nom": "nom2", "cognoms": "cognoms2", "es_admin": "0"}]

b) Action View

⇒ Request

- URL: http://localhost/yii/reserves/api/usuaris/{id}
- Method: GET
- Headers: X_USERNAME: username

  X_PASSWORD: password_encriptat

⇒ Response

- Return: Devuelve el usuario correspondiente al id indicado en la URL. Si el usuario no existe o no lo puede ver, devuelve error 404.
- JSON:

  {"id": "2", "nom_usuari": "user2", "contrasenya": "password2", "nom": "nom2", "cognoms": "cognoms2", "es_admin": "0"}
- Errors:

{"codError": 404, "descError": "VIEW: No existeix o no es pot accedir a l'usuari amb id=100"}

c) Action Create

⇒ Request

- URL: http://localhost/yii/reserves/api/usuaris/
- Method: POST
- Headers: Content-Type: application/x-www-form-urlencoded
  X_USERNAME: username
  X_PASSWORD: password_encriptat
- Data:

  nom_usuari=username&contrasenya=password_ecriptat&nom=nom&cognoms=cognom1+cognom2&es_admin=0_o_1

⇒ Response

- Return: Devuelve id del usuario creado. Si el usuario conectado no es admin, devuelve error 403.
- JSON:

  {"id": "11"}
- Errors:

  {"codError": 403, "descError": "CREATE: L'usuari no té permís per crear usuaris"}

d) Action Update

⇒ Request

- URL: http://localhost/yii/reserves/api/usuaris/{id}
- Method: PUT
- Headers: Content-Type: application/json
  X_USERNAME: username
  X_PASSWORD: password_encriptat

Data: {nom: “nou_nom”, es_admin: “1”} , etc
o bien {espai_id: “2”}, para dar permiso usuario-espacio

⇒ Response

- Return: Devuelve id del usuario actualizado. Si el usuario conectado no es admin, devuelve error 403. Si no existe el usuario a actualizar, devuelve error 404.
- JSON:

  {"id": "11"}
- Errors:

  {"codError": 403, "descError: "UPDATE: L'usuari no té permís per actualitzar usuaris"}

  {"codError": 404, "descError": "UPDATE: No existeix l’usuari amb id=100"}

e) Action Delete

⇒ Request

- URL: http://localhost/yii/reserves/api/usuaris/{id}
- Method: DELETE
- Headers: X_USERNAME: username
  X_PASSWORD: password_encriptat

⇒ Response

- Return: Devuelve id del usuario borrado. Si el usuario conectado no es admin, devuelve error 403. Si no existe el usuario a borrar, devuelve error 404.
- JSON:

{"id": "11"}

- Errors:

{"codError": 403, "descError: "DELETE: L'usuari no té permís per esborrar usuaris"}

{"codError": 404, "descError": "DELETE: No existeix l'usuari amb id=100"}

3. Model Reserves

f) Action List

⇒ Request

- URL: http://localhost/yii/reserves/api/reserves
- Filters: ?inici=aaaammddhh&fi=aaaammddhh&espai=1&usuari=1
- Method: GET
- Headers: X_USERNAME: username
  X_PASSWORD: password_encriptat

⇒ Response

- Return: Devuelve las reservas que puede ver el usuario conectado. Si el usuario conectado es admin, puede ver los datos del usuario de la reserva. Si no, se muestra null em "usuari_id".
- JSON:

[{"id": "2","data_hora": "2014-05-13 11:00:00",
"espai_id": {"id": "2","codi": "B","descripcio": "Espai B"},
"usuari_id": {"id": "2","nom_usuari": "user2","contrasenya": "password2","nom": "nom2","cognoms": "cognoms2","es_admin": "0"},
"data_modif": "2014-05-13 22:22:26"},
{"id": "3", "data_hora": "2014-06-13 12:00:00",
"espai_id": {"id": "3", "codi": "C", "descripcio": "Espai C"},
"usuari_id": {"id": "2", "nom_usuari": "user2", "contrasenya": "password2",
"nom": "nom2", "cognoms": "cognoms2", "es_admin": "0"},
"data_modif": "2014-05-17 17:55:52"}]

g) Action View

⇒ Request

- URL: http://localhost/yii/reserves/api/reserves/{id}
- Method: GET
- Headers: X_USERNAME: username

X_PASSWORD: password_encriptat

⇒ Response

- Return: Devuelve la reserva correspondiente al id indicado en la URL. Si la reserva no existe, devuelve error 404. Si el usuario conectado es admin, puede ver lo datos del usuario de la reserva. Si no, se muestra null en "usuari_id".
- JSON:

{"id": "2", "data_hora": "2014-05-13 11:00:00",
"espai_id": {"id": "2", "codi": "B", "descripcio": "Espai B"},
"usuari_id": {"id": "2","nom_usuari": "user2","contrasenya": "password2",
"nom": "nom2", "cognoms": "cognoms2", "es_admin": "0"},
"data_modif": "2014-05-13 22:22:26"}

- Errors:

{"codError": 404, "descError": "VIEW: No existeix la reserva amb id=100"}

h) Action Create

⇒ Request

- URL: http://localhost/yii/reserves/api/reserves/
- Method: POST
- Headers: Content-Type: application/x-www-form-urlencoded
  X_USERNAME: username
  X_PASSWORD: password_encriptat
- Data: data_hora=aaaammddhh&espai_id=2&usuari_id=2

⇒ Response

- Return: Devuelve id de la reserva creada. Si el usuario conectado no es admin, se guarda el propio usario.
- JSON:

{"id": "11"}

i) Action Update

⇒ Request

- URL: http://localhost/yii/reserves/api/reserves/{id}
- Method: PUT
- Headers: Content-Type: application/json
  X_USERNAME: username
  X_PASSWORD: password_encriptat

Data: {data_hora: “aaaammddhh”, espai_id: “2”, usuari_id: ”2”}

⇒ Response

- Return: Devuelve id de la reserva actualizado. Si el usuario conectado no es admin, devuelve error 403. Si no existe la reserva a actualizar, devuelve error 404.
- JSON:

  {"id": "11"}
- Errors:

  {"codError": 403, "descError: "UPDATE: L'usuari no té permís per actualitzar reserves"}

  {"codError": 404, "descError": "UPDATE: No existeix la reserva amb id=100"}

j) Action Delete

⇒ Request

- URL: http://localhost/yii/reserves/api/reserves/{id}
- Method: DELETE
- Headers: X_USERNAME: username
  X_PASSWORD: password_encriptat

⇒ Response

- Return: Devuelve id de la reserva borrada. Si el usuario conectado no es admin, devuelve error 403. Si no existe la reserva a borrar, devuelve error 404.
- JSON:

{"id": "11"}

- Errors:

{"codError": 403, "descError: "DELETE: L'usuari no té permís per esborrar reserves"}

{"codError": 404, "descError": "DELETE: No existeix la reserva amb id=100"}