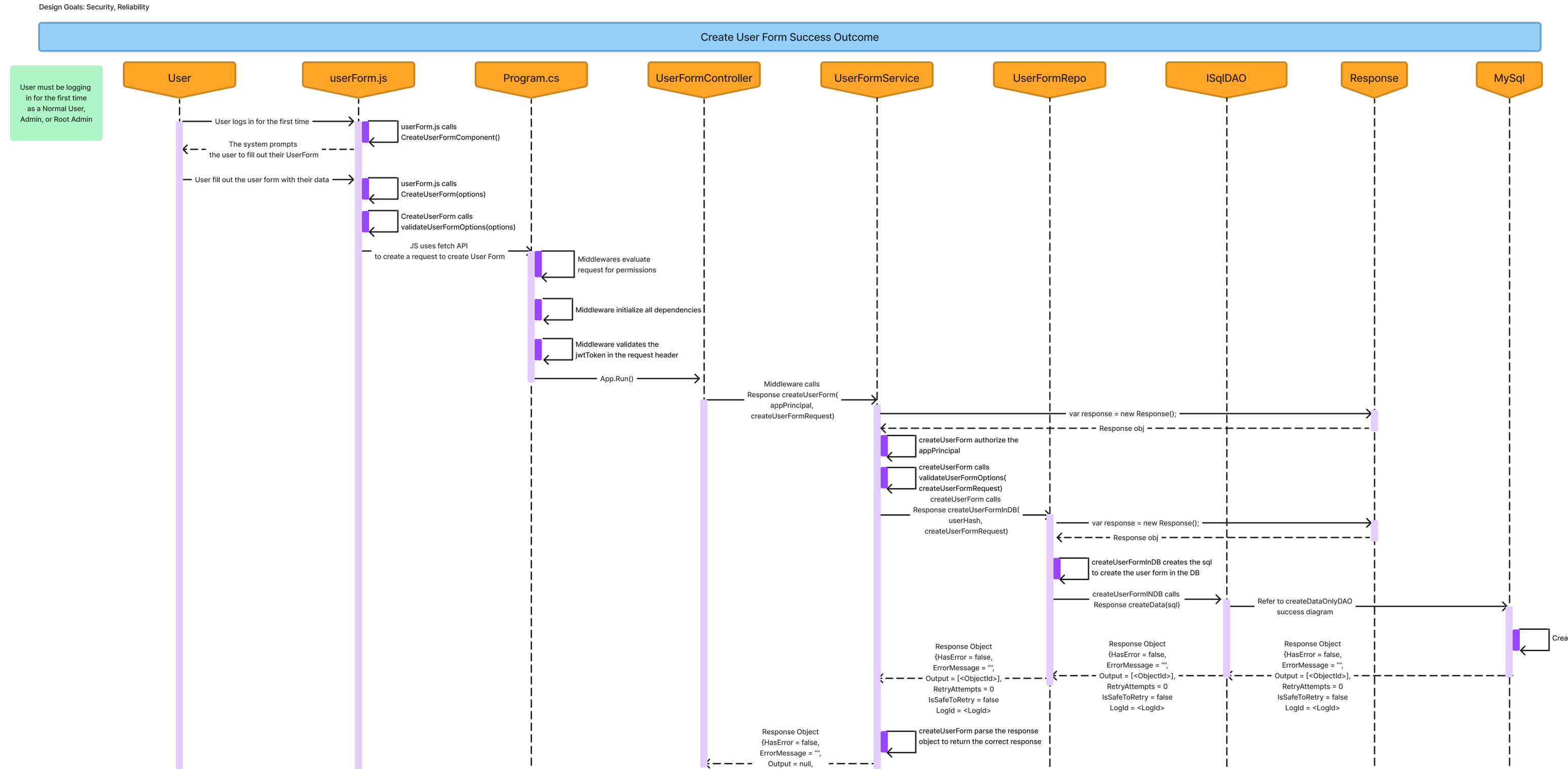
User Form

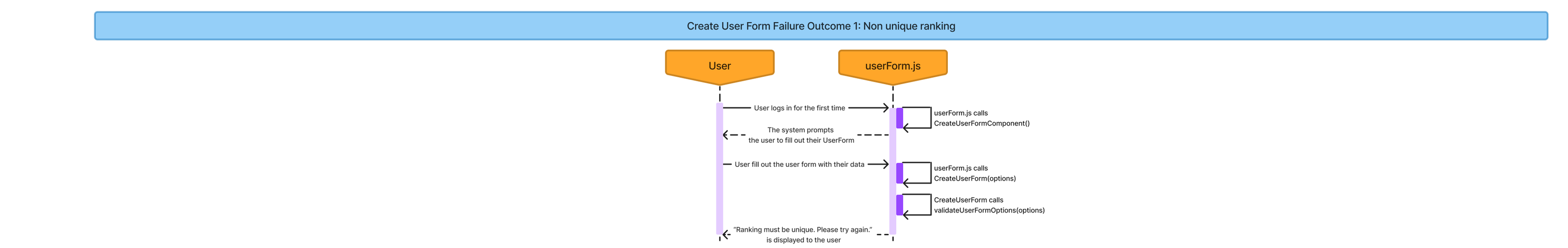
UF1 User Story: Create User Form

Design Goals: Security, Reliability



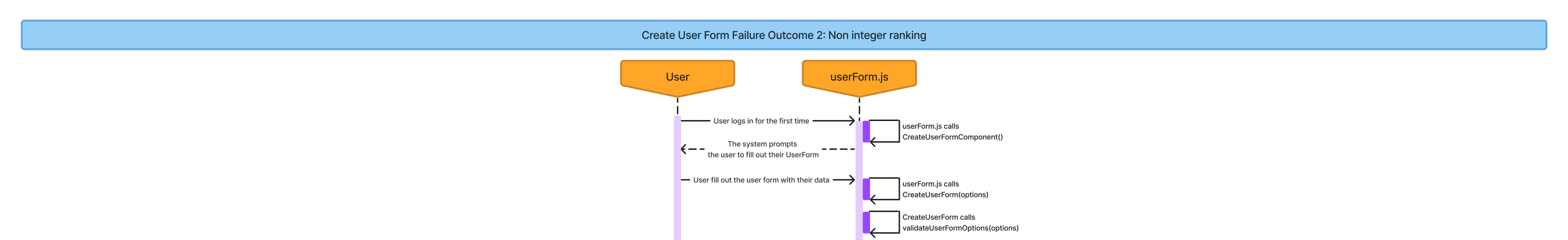
Create User Form Success Outcome

User
userForm.js
Program.cs
UserFormController
UserFormService
UserFormRepo
ISqlDAO
Response
MySql
User must be logging in for the first time as a Normal User, Admin, or Root Admin
User logs in for the first time
userForm.js calls CreateUserFormComponent()
The system prompts the user to fill out their UserForm
User fill out the user form with their data
userForm.js calls CreateUserForm(options)
CreateUserForm calls validateUserFormOptions(options)
JS uses fetch API to create a request to create User Form
Middlewares evaluate request for permissions
Middleware initialize all dependencies
Middleware validates the jwtToken in the request header
App.Run()
Middleware calls Response createUserForm(appPrincipal, createUserFormRequest)
var response = new Response();
Response obj
createUserForm authorize the appPrincipal
createUserForm calls validateUserFormOptions(createUserFormRequest)
createUserForm calls Response createUserFormInDB(userHash, createUserFormRequest)
var response = new Response();
Response obj
createUserFormInDB creates the sql to create the user form in the DB
createUserFormInDB calls Response createData(sql)
Refer to createDataOnlyDAO success diagram
Create User Form in DB
Response Object (HasError = false, ErrorMessage = "", Output = [<ObjectId>], RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Response Object (HasError = false, ErrorMessage = "", Output = [<ObjectId>], RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Response Object (HasError = false, ErrorMessage = "", Output = [<ObjectId>], RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
createUserForm parse the response object to return the correct response
Response Object (HasError = false, ErrorMessage = "", Output = null, RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Refer to logging sequence diagram
return Ok(Response)
"The User Form is successfully saved" is displayed to the user. The user is automatically move to the LLI Management Page



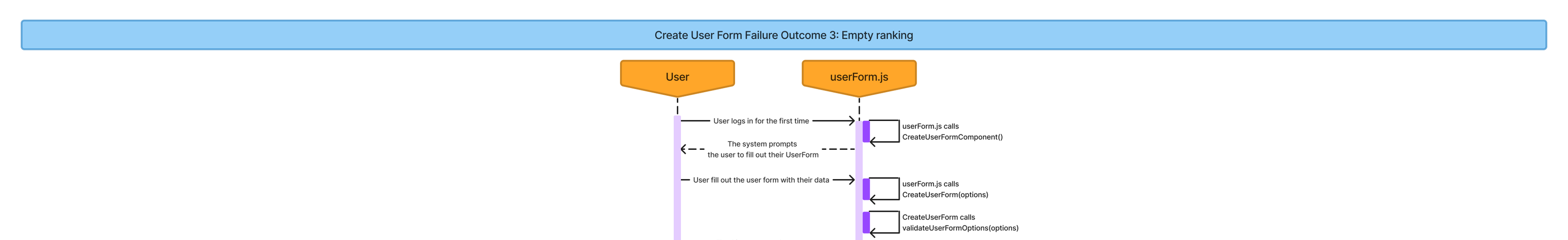
Create User Form Failure Outcome 1: Non unique ranking

User
userForm.js
User logs in for the first time
userForm.js calls CreateUserFormComponent()
The system prompts the user to fill out their UserForm
User fill out the user form with their data
userForm.js calls CreateUserForm(options)
CreateUserForm calls validateUserFormOptions(options)
"Ranking must be unique. Please try again." is displayed to the user



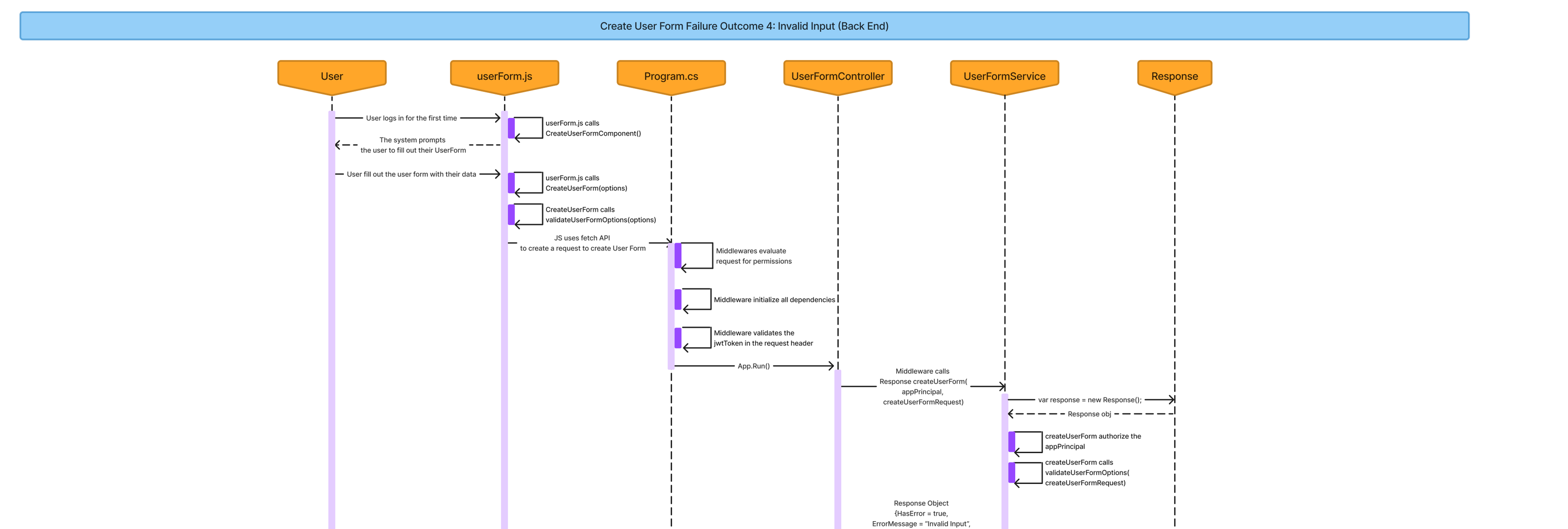
Create User Form Failure Outcome 2: Non integer ranking

User
userForm.js
User logs in for the first time
userForm.js calls CreateUserFormComponent()
The system prompts the user to fill out their UserForm
User fill out the user form with their data
userForm.js calls CreateUserForm(options)
CreateUserForm calls validateUserFormOptions(options)
"Ranking must be between 1 and 10. Please try again." is displayed to the user



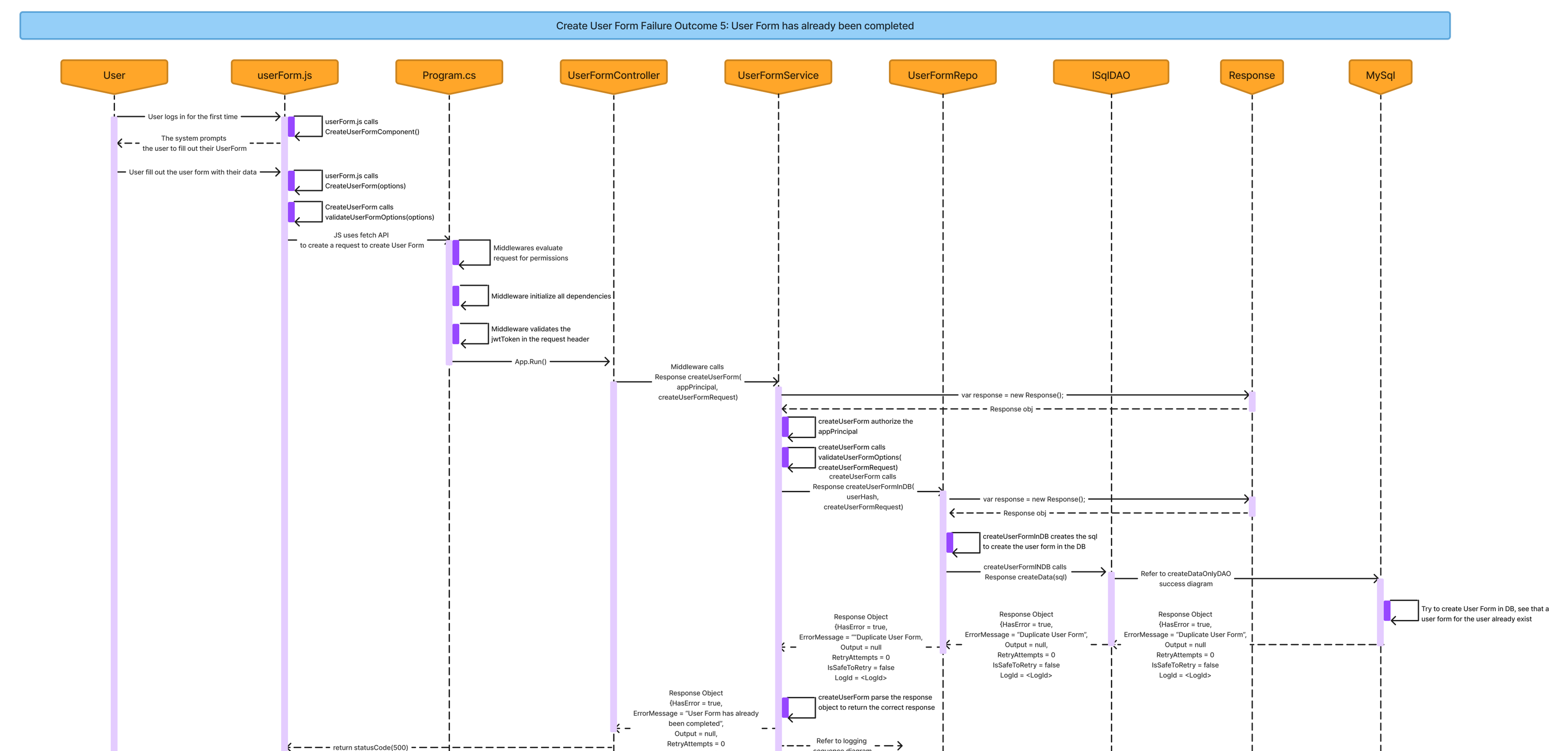
Create User Form Failure Outcome 3: Empty ranking

User
userForm.js
User logs in for the first time
userForm.js calls CreateUserFormComponent()
The system prompts the user to fill out their UserForm
User fill out the user form with their data
userForm.js calls CreateUserForm(options)
CreateUserForm calls validateUserFormOptions(options)
"Ranking must not be empty. Please try again." is displayed to the user



Create User Form Failure Outcome 4: Invalid Input (Back End)

User
userForm.js
Program.cs
UserFormController
UserFormService
Response
User logs in for the first time
userForm.js calls CreateUserFormComponent()
The system prompts the user to fill out their UserForm
User fill out the user form with their data
userForm.js calls CreateUserForm(options)
CreateUserForm calls validateUserFormOptions(options)
JS uses fetch API to create a request to create User Form
Middlewares evaluate request for permissions
Middleware initialize all dependencies
Middleware validates the jwtToken in the request header
App.Run()
Middleware calls Response createUserForm(appPrincipal, createUserFormRequest)
var response = new Response();
Response obj
createUserForm authorize the appPrincipal
createUserForm calls validateUserFormOptions(createUserFormRequest)
Response Object (HasError = true, ErrorMessage = "Invalid Input", Output = null, RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Refer to logging sequence diagram
return statusCode(400)
"Bad Request" is displayed to the user.

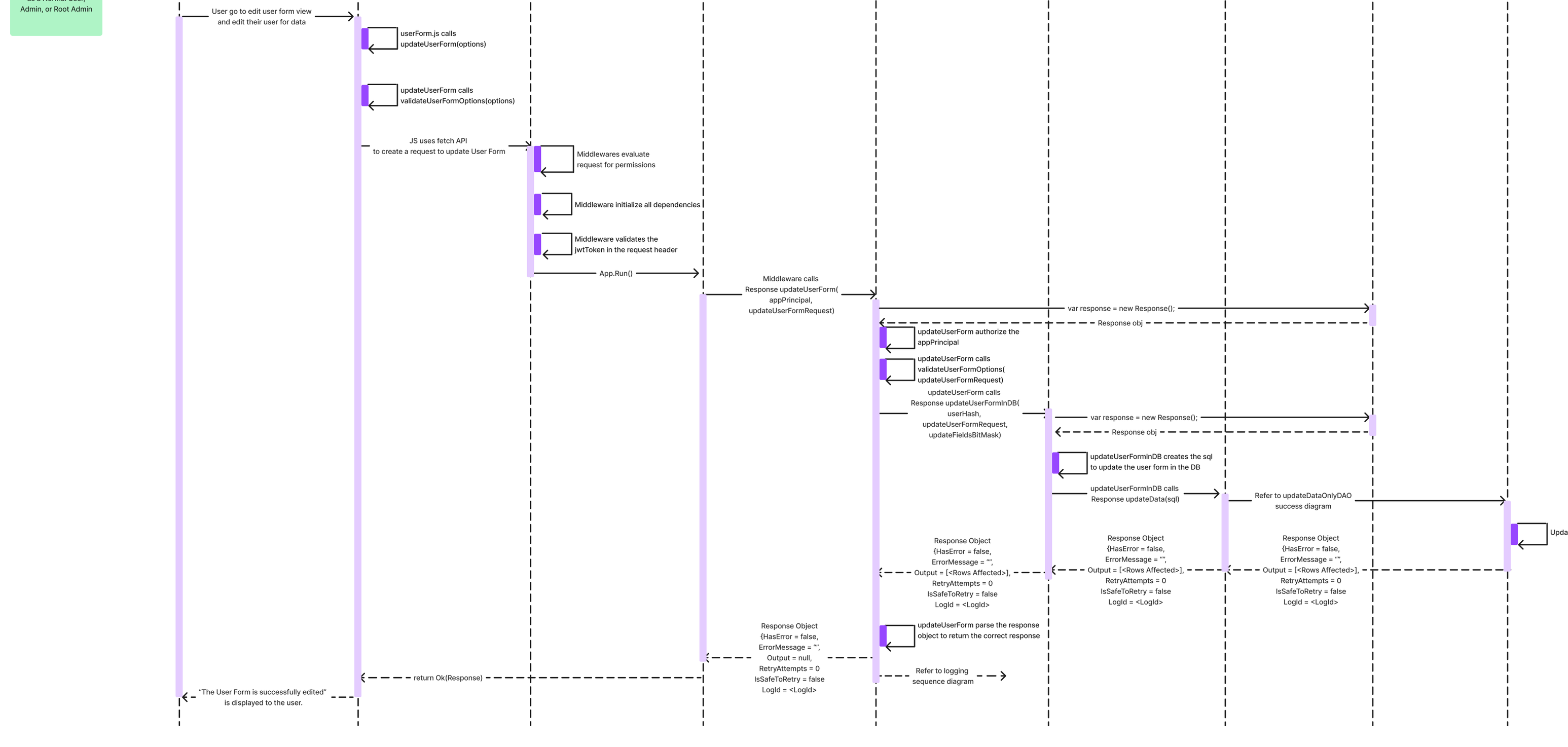


Create User Form Failure Outcome 5: User Form has already been completed

User
userForm.js
Program.cs
UserFormController
UserFormService
UserFormRepo
ISqlDAO
Response
MySql
User logs in for the first time
userForm.js calls CreateUserFormComponent()
The system prompts the user to fill out their UserForm
User fill out the user form with their data
userForm.js calls CreateUserForm(options)
CreateUserForm calls validateUserFormOptions(options)
JS uses fetch API to create a request to create User Form
Middlewares evaluate request for permissions
Middleware initialize all dependencies
Middleware validates the jwtToken in the request header
App.Run()
Middleware calls Response createUserForm(appPrincipal, createUserFormRequest)
var response = new Response();
Response obj
createUserForm authorize the appPrincipal
createUserForm calls validateUserFormOptions(createUserFormRequest)
createUserForm calls Response createUserFormInDB(userHash, createUserFormRequest)
var response = new Response();
Response obj
createUserFormInDB creates the sql to create the user form in the DB
createUserFormInDB calls Response createData(sql)
Refer to createDataOnlyDAO success diagram
Try to create User Form in DB, see that a user form for the user already exist
Response Object (HasError = true, ErrorMessage = ""Duplicate User Form", Output = null RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Response Object (HasError = true, ErrorMessage = "Duplicate User Form", Output = null, RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Response Object (HasError = true, ErrorMessage = "Duplicate User Form", Output = null RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
createUserForm parse the response object to return the correct response
Response Object (HasError = true, ErrorMessage = "User Form has already been completed", Output = null, RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Refer to logging sequence diagram
return statusCode(500)
"The User Form has already been completed. Please contact an admin if the issue persist." is displayed to the user.

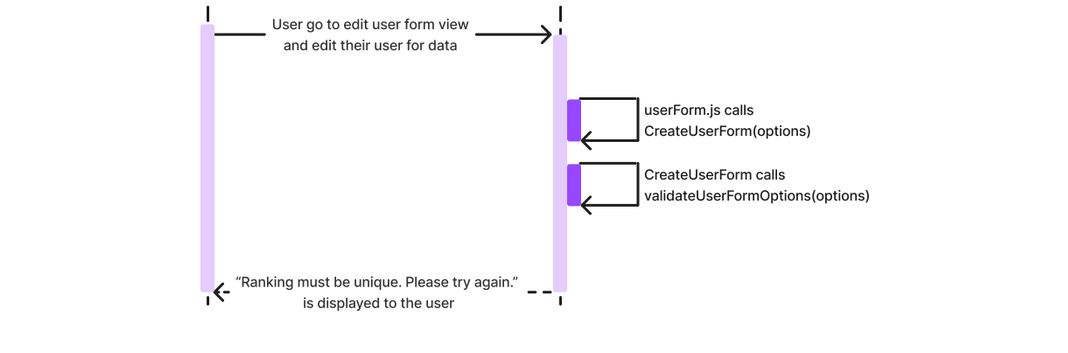
UF2 User Story: Edit User Form

Design Goals: Security, Reliability



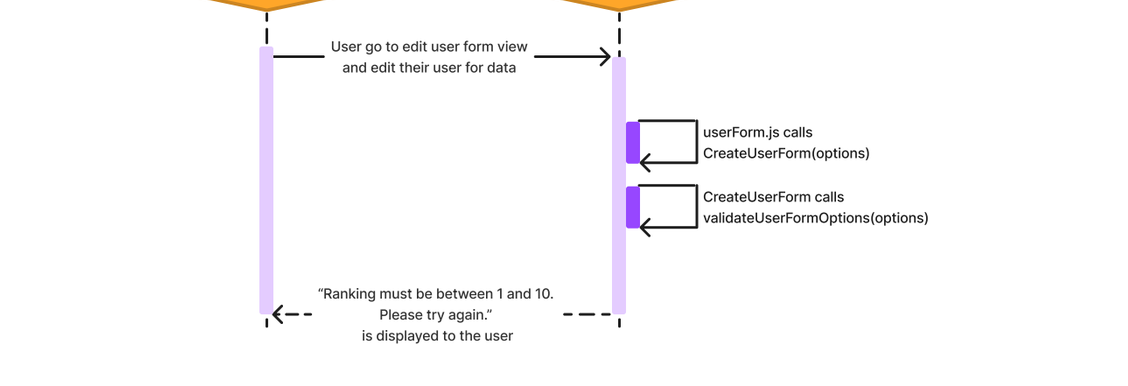
Edit User Form Success Outcome

User
userForm.js
Program.cs
UserFormController
UserFormService
UserFormRepo
ISqlDAO
Response
MySql
User must be logged in as a Normal User, Admin, or Root Admin
User go to edit user form view and edit their user for data
userForm.js calls updateUserForm(options)
updateUserForm calls validateUserFormOptions(options)
JS uses fetch API to create a request to update User Form
Middlewares evaluate request for permissions
Middleware initialize all dependencies
Middleware validates the jwtToken in the request header
App.Run()
Middleware calls Response updateUserForm(appPrincipal, updateUserFormRequest)
var response = new Response();
Response obj
updateUserForm authorize the appPrincipal
updateUserForm calls validateUserFormOptions(updateUserFormRequest)
updateUserForm calls Response updateUserFormInDB(userHash, updateUserFormRequest, updateFieldsBitMask)
var response = new Response();
Response obj
updateUserFormInDB creates the sql to update the user form in the DB
updateUserFormInDB calls Response updateData(sql)
Refer to updateDataOnlyDAO success diagram
Update User Form in DB
Response Object (HasError = false, ErrorMessage = "", Output = [<Rows Affected>], RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Response Object (HasError = false, ErrorMessage = "", Output = [<Rows Affected>], RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Response Object (HasError = false, ErrorMessage = "", Output = [<Rows Affected>], RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
updateUserForm parse the response object to return the correct response
Response Object (HasError = false, ErrorMessage = "", Output = null, RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Refer to logging sequence diagram
return Ok(Response)
"The User Form is successfully edited" is displayed to the user.



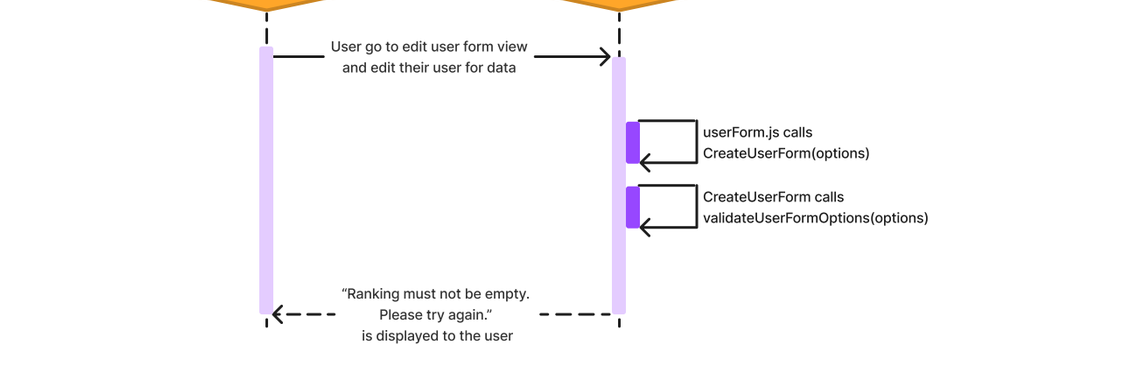
Edit User Form Failure Outcome 1: Non unique ranking

User
userForm.js
User go to edit user form view and edit their user for data
userForm.js calls CreateUserForm(options)
CreateUserForm calls validateUserFormOptions(options)
"Ranking must be unique. Please try again." is displayed to the user



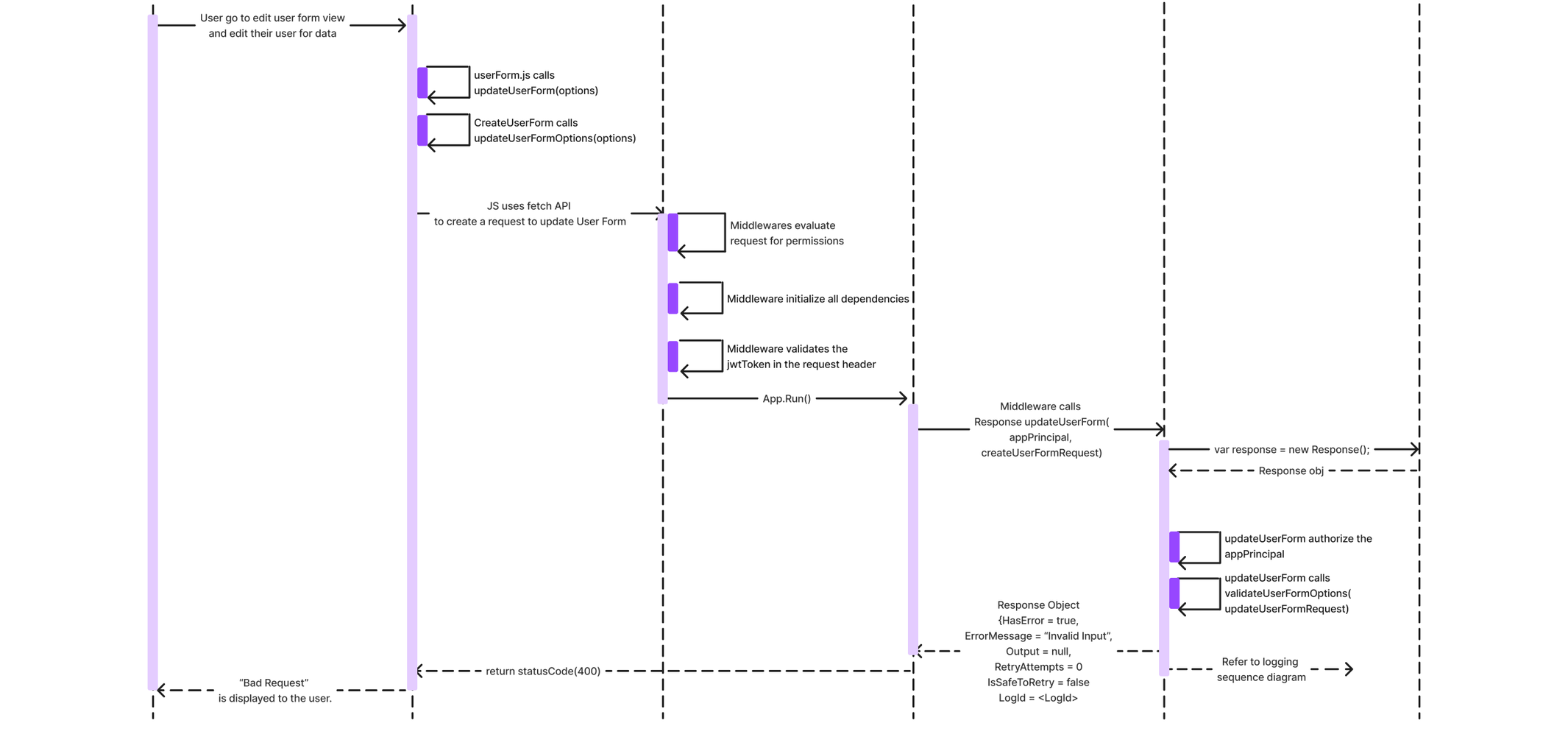
Edit User Form Failure Outcome 2: Non integer ranking

User
userForm.js
User go to edit user form view and edit their user for data
userForm.js calls CreateUserForm(options)
CreateUserForm calls validateUserFormOptions(options)
"Ranking must be between 1 and 10. Please try again." is displayed to the user



Edit User Form Failure Outcome 3: Empty ranking

User
userForm.js
User go to edit user form view and edit their user for data
userForm.js calls CreateUserForm(options)
CreateUserForm calls validateUserFormOptions(options)
"Ranking must not be empty. Please try again." is displayed to the user



Edit User Form Failure Outcome 4: Invalid Input (Back End)

User
userForm.js
Program.cs
UserFormController
UserFormService
Response
User go to edit user form view and edit their user for data
userForm.js calls updateUserForm(options)
CreateUserForm calls updateUserFormOptions(options)
JS uses fetch API to create a request to update User Form
Middlewares evaluate request for permissions
Middleware initialize all dependencies
Middleware validates the jwtToken in the request header
App.Run()
Middleware calls Response updateUserForm(appPrincipal, createUserFormRequest)
var response = new Response();
Response obj
updateUserForm authorize the appPrincipal
updateUserForm calls validateUserFormOptions(updateUserFormRequest)
Response Object (HasError = true, ErrorMessage = "Invalid Input", Output = null, RetryAttempts = 0 IsSafeToRetry = false LogId = <LogId>
Refer to logging sequence diagram
return statusCode(400)
"Bad Request" is displayed to the user.