Integrated Learning System: Test Cases

API Test Cases

Evidence of API testing with Postman are recorded in the file `Integrated Learning System V1.postman_collection.json`.

Test Case ID	Test Case	Relative Route	Preconditions	Request	Expected Response	Actual Response	Verdict
1. Server Health Check							
1.1	Health check request when server under good condition and running under *development* mode	/api/ping	Database is on; *config.edn* file: `:env` key assigned with `:dev`	GET; JSON request body as shown in Figure 1 of Appendix D	HTTP Code 200; Content Type: JSON; Body has fields: - message: "Server is healthy" - errors: `empty list` - `debugInfos`	As expected	Passed
1.2	Health check request when server running under *development* mode and database is not turned on		Database is not on; *config.edn* file: *same arrangement as of test case 1.1*	*Same request as test case 1.1*	HTTP Code 200; Content Type: JSON; Body has fields: - message: "Server is not ready for request" - errors: `a list containing the object` `{"area": "database", "message": "Failed to establish connection with database."}` - `debugInfos`	As expected; Demonstrated by Figure 2	Passed

1.3	Health check request when server under good condition and running under *production* mode		Database is on; *config.edn* file: *:env* key assigned with *:prod*	*Same request as test case 1.1*	Same as test case 1.1, but without the field debugInfos in response body	As expected	Passed
1.4	Health check request when server running under *development* mode and database is not turned on	/api/ping	Database is not on; *config.edn* file: *same arrangement as of test case 1.3*	*Same request as test case 1.1*	Same as test case 1.2, but without the field debugInfos in response body	As expected	Passed
2. Accounts							

2.1	Request to create account with invalid role	/api/v1/accounts/	*none*	POST request; JSON request body:
				
				`{`
				`    "role": "moderator",`
				`    "username": "moderator",`
				`    "password": "todo-hashed-password",`
				`    "firstName": "John the moderator",`
				`    "lastName": "Doe",`
				`    "dateOfBirth": "01/01/1980",`
				`    "personalEmail": "john_doe.moderator@domain.com",`
					HTTP Code 401; Content Type: JSON; Response body:
					
					`{`
					`    "title": "Invalid account registration request for username 'moderator'.",`
					`    "errors": {`
					`        "role": [`
					`            "Invalid role value: 'moderator'. Allowed values: 'admin', 'teacher', 'student'."`
					`        ]`
					`    }`
					`}`
						As expected	Passed

				"phoneNumber": "+849041234568" }			

2.2.1	Request to create teacher account with invalid data	/api/v1/accounts/	*none*	POST request;
				JSON request body:
				{
				    "role": "teacher",
				    "username": "",
				    "password": "",
				    "firstName": "",
				    "lastName": "",
				    "dateOfBirth": "01/01/1980",
				    "personalEmail": "",
				    "phoneNumber": ""
				}
					HTTP Code 422;
					Content Type: JSON;
					Response body:
					{
					 "title": "Validation errors.",
					 "errors": {
					  "phoneNumber": [
					   "phoneNumber must not be blank."
					  ],
					  "password": [
					   "password must not be blank."
					  ],
					  "username": [
					   "username must not be blank."
					  ],
					  "personalEmail": [
					   "personalEmail must not be blank."
					  ],
					  "lastName": [
					   "lastName must not be blank."
					  ],
					  "firstName": [
					   "firstName must not be blank."
					  ]
					 }
					}
						As expected	Passed

2.2.2				POST request;	HTTP Code 422;	As expected	Passed

				JSON request body:	Content Type: JSON; Response body:		

				{
				 "role": "teacher",
				 "username": "t*a😊cer",
				 "password": "todo-hashed-password",
				 "firstName": "selct * from teacher where account_id = 00000000-0000-0000-0000-000000000000",
				 "lastName": "",
				 "dateOfBirth": "11/01/1980",
				 "personalEmail": "john_doe@teacher@domain.com",
				 "phoneNumber": "(081)9041234568"
				}
					{
					 "title": "Validation errors.",
					 "errors": {
					 "phoneNumber": [
					 "Invalid phone number value for phoneNumber: '(081)9041234568'."
					],
					 "username": [
					 "username can only contain alphabetic, numeric, '_', '.' or '-' characters."
					],
					 "personalEmail": [
					 "Invalid email value for personalEmail: 'john_doe@teacher@domain.com'."
					],
					 "lastName": [
					 "lastName must not be blank."
					],
					 "firstName": [
					 "firstName's length must not exceed 50."
					]
					 }
					}		

2.3	Request to create teacher	/api/v1/accounts/	*none*	POST request; JSON request body:	HTTP Code 401; Content Type: JSON;	As expected	Passed

	account with unsupported date format for `dateOfBirth` field			`{ "role": "teacher", "username": "t*a😊cer", "password": "todo-hashed-password", "firstName": "selct * from teacher where account_id = 00000000-0000-0000-0000-000000000000", "lastName": "", "dateOfBirth": "01/1980", "personalEmail": "john_doe@teacher@domain.com", "phoneNumber": "(081)9041234568" }`	Response body: `{ "title": "Invalid account registration request for username 't*a😊cer'.", "errors": { "dateOfBirth": [ "Invalid date value for dateOfBirth: '01/1980'. Allowed formats: dd/MM/yyyy, yyyy-MM-dd" ] } }`		
2.4	Request to create teacher account with valid request body	/api/v1/accounts/	Database is on; There are NO teachers with username "teacher" existed in the database	POST request; JSON request body: `{ "role": "teacher", "username": "john_doe.teacher", "password": "todo-hashed-password", "firstName": "John the Teacher", "lastName": "Nguyễn", "dateOfBirth": "01/01/1980",`	HTTP Code 201; Header: has entry "Location" with value "/teachers/john_doe.teacher"	As expected	Passed

					`"personalEmail": "john_doe.teacher@domain.com",`
					`"phoneNumber": "+849041234568"`
					`}`		

2.5	Request to attempt creating duplicate teacher account	/api/v1/accounts/	Database is on; There is already a teacher account with username "teacher" existed in the database	*same as test case 2.4*	HTTP Code 422; Content Type: JSON; Response body:
					`{`
					`  "title": "Data duplication",`
					`  "errors": {`
					`    "username": "Account with username [john_doe.teacher] already exists."`
					`  }`
					`}`
						As expected	Passed

2.6	Request to create teacher when database is offline	/api/v1/accounts/	Database is turned off	*same as test case 2.4*	HTTP Code 302; Header: entry "Location" with value "/api/ping"	HTTP Code 500; Response body:
						`{`
						`  "type": "exception",`
						`  "class": "clojure.lang.ExceptionInfo"`
						`}`
							Failed

2.7	Request to create student with valid request body	/api/v1/accounts/	Database is on; There are NO students with username "student" existed in the database	POST request; JSON request body:
				`{`
				`  "role": "student",`
				`  "username": "student",`
					HTTP Code 201; Header: entry "Location" with value "/students/student"	As expected	Passed

				"password": "hashed-password",
				  "firstName": "Tommy the Student",
				  "lastName": "Hoàng",
				  "dateOfBirth": "21/12/1996",
				  "personalEmail": "tommy_learner_for_life@domain.com",
				  "phoneNumber": "+84904123457"
				}			

3. Teachers							

3.1	Request to get all teachers	/api/v1/teachers/	Database is online;
			Test case 2.4 completed successfully
				GET request;
				Empty body
					HTTP Code 200;
					Content Type: JSON;
					Response body:
					[
					  {
					   "username": "teacher",
					   "firstName": "John the Teacher",
					   "lastName": "Nguyễn",
					   "personalEmail": "john_doe.teacher@domain.com",
					   "dateOfBirth": "01/01/1980",
					   "phoneNumber": "+849041234568",
					   "fullName": "John the Teacher Nguyễn",
					   "displayName": "John the Teacher Nguyễn (teacher)"
					  }
						As expected	Passed

					]		
3.2	Request to get all teachers when database is offline		Database is turned off		HTTP Code 302; Header: entry "Location" with value "/api/ping"	As expected	Passed
4. Students							
4.1	Request to get all students	/api/v1/studenets/	Database is online; *Test case 2.7* completed successfully	GET request; Empty body	HTTP Code 200; Content Type: JSON; Response body: ```[{ "username": "student", "firstName": "Tommy the Student", "lastName": "Hoàng", "personalEmail": "tommy_learner_for_life@domain.com", "dateOfBirth": "21/12/1996", "phoneNumber": "+84904123457", "fullName": "Tommy the Student Hoàng", "displayName": "Tommy the Student Hoàng (student)" }]```	As expected	Passed
4.2	Request to get all students when database is offline		Database is turned off		HTTP Code 302; Header: entry "Location" with value "/api/ping"	As expected	Passed
5.Timeslots							

5.1	Valid request to create the first timeslot	/api/v1/time slots/	Database is on; There are no timeslot records yet	POST request; JSON request body:
				
				`{`
				`  "startAt": "07:30",`
				`  "durationMins": 90`
				`}`
					HTTP Code 201; Header: entry "Location" with value "/timeslots/1"; Content Type: JSON; Response body:
					
					`{`
					`  "number": 1,`
					`  "startAt": "07:30",`
					`  "durationMins": 90,`
					`  "uri":`
					`"/timeslots/1"`
					`}`
						As expected	Passed

5.2	Valid request to append new non-overlapping timeslot that comes after the first timeslot		Database is on; *Test case 5.1* executed successfully	POST request; JSON request body:
				
				`{`
				`  "startAt": "09:10",`
				`  "durationMins": 90`
				`}`
					HTTP Code 201; Header: entry "Location" with value "/timeslots/2"; Content Type: JSON; Response body:
					
					`{`
					`  "number": 2,`
					`  "startAt": "09:10",`
					`  "durationMins": 90,`
					`  "uri":`
					`"/timeslots/2" }`
						As expected	Passed

5.3	Request to add new timeslot that overlaps with the first timeslot	/api/v1/time slots/	Database is on; *Test case 5.1* executed successfully	POST request; JSON request body:
				
				`{`
				`  "startAt": "07:00",`
				`  "durationMins": 45`
				`}`
					HTTP Code 422; Content Type: JSON; Response body:
					
					`{`
					`  "title": "Data conflict",`
					`  "description": "Timeslot to add overlaps with timeslot 1"`
					`}`
						HTTP Code 201; Content Type: JSON; Header: has entry "Location" with value "/timeslots/"; Response body:
						
						`{`
						`  "startAt": null,`
						`  "uri": "/timeslots/"`
						`}`
							Failed

5.4	Request to create new timeslot with invalid body		*none*	POST request; JSON request body:
				
				``
				{
				 "durationMins": 48.5,
				 "startAt": "7:00"
				}
				``
					HTTP Code 401; Content Type: JSON; Response body:
					
					``
					{
					 "title": "Invalid time-slot insertion request body.",
					 "errors": {
					 "durationMins": [
					 "durationMins value is expected to be a positive integer, but found: 48.5"
					],
					 "startAt": [
					 "Invalid time value for startAt: '7:00'. Expected format: HH:mm"
					]
					 }
					}
					``
						As expected	Passed

6. Courses							

6.1	Request to add a course	/api/v1/cour ses/	Database is on; There are NO course record with code column value "LEARNING101"	POST request; JSON request body:
				
				``
				{
				 "courseCode": "LEARNING101",
				 "courseName": "Learning 101"
				}
				``
					HTTP Code 201; Content Type: JSON; Header: has "Location" entry with value "/courses/LEARNING101"; Response body:
					
					``
					{
					 "courseCode": "LEARNING101",
					 "courseName": "Learning 101",
					 "description": null,
					``
						As expected	Passed

					`"status": "ACTIVE",`
					`"uri":`
					`"/courses/LEARNING101`
					`"`
					`}`		

6.2	Request to attempt adding duplicate course	/api/v1/cour ses/	Database is on;
			Test case 6.1 executed successfully
				same as test case 6.1	HTTP Code 422;
					Content Type: JSON;
					Response body:
					`{`
					`"title": "Data`
					`conflicts.",`
					`"errors": {`
					`"courseCode":`
					`"Course with code`
					`[LEARNING101] already`
					`exists."`
					`}`
					`}`
						As expected	Passed

6.3	Invalid request to add new course with body has wrong field type	/api/v1/cour ses/	*none*	POST request;
				JSON request body:
				`{`
				`"courseCode": 101,`
				`"courseName": "Learn`
				`ing 101 but code is`
				`number"`
				`}`
					HTTP Code 401;
					Content Type: JSON;
					Response body:
					`{`
					`"title": "Invalid`
					`request",`
					`"errors": {`
					`"courseCode":`
					`"Expected a string`
					`value, but found a`
					`number value."`
					`}`
					`}`
						HTTP Code 500;
						Content Type: JSON;
						Response body:
						`{`
						`"title": "Failed to`
						`process this request`
						`for creating`
						`course.",`
						`"errors": null`
						`}`
							Failed

7. Classes							
7.1	Request to create a class from existing course	/api/v1/clas ses/	Database is on; *Test case 6.1 executed successfully*	POST request; JSON request body: `{`	HTTP Code 201; Header: has entry "Location" with value "/classes/CLASS01" Content Type: JSON;	As expected	Passed

				`"courseCode": "LEARNING101",`
				`"className": "CLASS01"`
				`}`
					Response body:
					`{`
					`"className":`
					`"CLASS01",`
					`"courseName":`
					`"Learning 101",`
					`"courseCode":`
					`"LEARNING101",`
					`"courseDescription":`
					`null,`
					`"courseStatus":`
					`"ACTIVE",`
					`"uri":`
					`"/classes/CLASS01"`
					`}`		

7.2	Request to attempt creating duplicate class	/api/v1/classes/	Database is on; *Test case 7.1* executed successfully	*same as test case 7.1*	HTTP Code 422; Content Type: JSON; Response body:
					`{`
					`"title": "Data conflicts.",`
					`"errors": {`
					`"className": [`
					`"Class name 'CLASS01' is already used."`
					`]`
					`}`
					`}`
						As expected	Passed

7.3	Request to create a class from non-existing course	/api/v1/classes/	Database is on; There are NO course record with code column value "ALGO101"	POST request;
				JSON request body:
				`{`
				`"courseCode": "ALGO101",`
				`"className": "CLAZZ01"`
				`}`
					HTTP Code 422; Content Type: JSON; Response body:
					`{`
					`"title": "Data conflicts.",`
					`"errors": {`
					`"courseCode": [`
						As expected	Passed

					``
					    "No courses of
					code 'ALGO101'
					found."
					    ]
					  }
					}
					``		

8. Class periods						

8.1	Request to query class periods with route having invalid path	/api/v1/classes/CLA^01/periods/	*none*	GET request; Empty body	HTTP Code 401; Content Type: JSON; Response body:
					``
					{
					  "title": "Invalid request",
					  "errors": {
					    "className": [
					      "className can only contain alphabetic, numeric, '_', '.' or '-' characters."
					    ]
					  }
					}
					``
						As expected	Passed

8.2	Request to query class periods with route referencing non-existing class	/api/v1/classes/CLAZZ01/periods/	Database is on; There are NO class record with class_name column value "CLAZZ01"		HTTP Code 200; Content Type: JSON; Response body: ``` [] ```	As expected	Passed
9. Class members						

9.1	Request to query class members with route having invalid path	/api/v1/classes/CL^!/members/	*none*	GET request; Empty body	HTTP Code 401; Content Type: JSON; Response body:
					``
					{
					  "title": "Invalid request",
					  "errors": {
					``
						As expected	Passed

					```json "className": ["className can only contain alphabetic, numeric, '_', '.' or '-' characters."] } } ```		
9.2	Request to query class members with route referencing non-existing class	/api/v1/clas ses/CLAZZ01/ members/			HTTP Code 200; Content Type: JSON; Response body: [ ]	As expected	Passed
9.3	Request to query class members with their timetable entries, but route queries are missing	/api/v1/clas ses/CLASS01/ members/time table	none	GET request; Empty body	HTTP Code 401; Content Type: JSON; Response body:	As expected	Passed
9.4	Request to query class members with their timetable entries, but route queries are in unsupported format	/api/v1/clas ses/CLASS01/ members/time table?from-date=01/11/2 022&to-date=01/12/2 022			```json { "title": "Invalid request", "errors": {} } ```	As expected	Passed

9.5	Valid request to query class members with their timetable entries	/api/v1/classes/CLASS01/members/timetable?from-date=2022-10-03&to-date=2022-10-05	*Test case 14.4.2 executed successfully*	GET request; Empty body	HTTP Code 200; Content Type: JSON; Response body:	As expected; Response body demonstrated by Figure 3	Passed

```
{
 "classTeacher": {
  "username": "john_doe.teacher",
  "firstName": "John the Teacher",
  "lastName": "Nguyễn",
  "fullName": "John the Teacher Nguyễn",
  "displayName": "John the Teacher Nguyễn (john_doe.teacher)",
  "timetable": [
   { "timeslotNumber": 1, "schoolDate": "2022-10-03", "className": "CLASS01", "courseCode": "LEARNING101", "courseName": "Learning 101" },
   { "timeslotNumber": 2, "schoolDate": "2022-10-03", "className": "CLASS01", "courseCode": "LEARNING101", "courseName": "Learning 101" },
   { "timeslotNumber": 2, "schoolDate": "2022-10-04", "className": "CLASS01", "courseCode": "LEARNING101", "courseName": "Learning 101" }
  ]
 },
 "classStudents": [
  {
   "username": "student",
   "firstName": "Tommy the Student",
   "lastName": "Hoàng",
   "fullName": "Tommy the Student Hoàng",
```

```
      "displayName": "Tommy
the Student Hoàng
(student)",
      "timetable": [
      { "timeslotNumber":
1, "schoolDate": "2022-
10-03", "className":
"CLASS01", "courseCode":
"LEARNING101",
"courseName": "Learning
101" },
      { "timeslotNumber":
2, "schoolDate": "2022-
10-03", "className":
"CLASS01", "courseCode":
"LEARNING101",
"courseName": "Learning
101" },
      { "timeslotNumber":
2, "schoolDate": "2022-
10-04", "className":
"CLASS01", "courseCode":
"LEARNING101",
"courseName": "Learning
101" }
      ]
      }
      ]
}
```

10. Miscellaneous							
10.1	Swagger specification should be generated	`/swagger.json`	*none*	GET request; Empty body	HTTP Code 200; Content Type: JSON; Response Body: Have fields "info" and "paths", each has meaningful values	As expected	Passed

C.2. Webpage Test Cases

Test Case ID	Test Case	Webpage Relative URL	Presumptions & Actions	Expected Behaviours	Actual Behaviours	Verdict
11. Homepage						
11.1.1	Homepage should be rendered correctly	/	Open the page in the browser	Navigation bar should be attached at the top of the page, and has the following options: - "Students" links to "/students" page - "Teachers" links to "/teachers" page - "Classes" links to "/classes/" page	As expected	Passed
11.1.2		/home			As expected	Passed
12. Classes Info Pages						
12.1	Page displaying all classes should be rendered correctly	/classes	• Execute *test case 7.1* successfully if not yet • Open this page in the browser	Class table should have 1 entry with: - "Class Name" of value to "CLASS01". The text links to "/classes/CLASS01" page - "Teacher Name (Teacher Username)" of value "<Not Assigned>", the text is italic. - "(Course Code) Course Name" of value "(LEARNING101) Learning 101"	As expected; Demonstrated by Figure 4	Passed
12.2	Info page for an existing class should be rendered correctly	/classes /CLASS01/	• *Test case 7.1* should have been executed successfully • Open this page in the web browser	The following text entries should be available: - "Class timetable" linking to page "/classes/CLASS01/ timetable" - "Manage class members" linking to page "/classes/CLASS01/ _actions/ manage-class-members". - "Organise class weekly schedule" linking to page "/classes/CLASS01/ _actions/ organise-weekly-schedule".	As expected; Demonstrated by Figure 5	Passed

12.3	Info page for class timetable should be rendered correctly	`/classes /CLASS01 /timetable`	• *Test case 14.4.2 should have been executed successfully* • Then, open this page under test in the web browser	- The "Previous week" and "Next week" buttons should be functional and directs the web browsers to the correct pages - Class periods *from 03-Oct-2022* to *03-Dec-2022* should be visible	As expected; Demonstrated by figures numbered from Figure 6 to Figure 11	Passed
		13. Class Action Page: Manage Class Members				
13.1	*Manage Class Member* page should not allow submission when no teacher is chosen yet		• Continues from the test steps of *test case 12.2* to get to this page • In the page: Click on the "Submit" button	Banner has colour changed to red, with message "ONE teacher must be assigned."	As expected; Demonstrated by Figure 12	Passed
13.2	*Manage Class Member* page should facilitate assigning teacher to class	`/classes/ CLASS01/ _actions/ manage- class- members`	• Continues from the test steps of *test case 13.1* • In the input field for teacher selection, type 't', then click on the entry "John the Teacher Nguyễn (john_doe.teacher)" from them dropdown menu. • Then, click the "Submit" button	Banner has colour changed to green, with message "Class members updated successfully"	As expected; Demonstrated by Figure 13	Passed
13.3	*Manage Class Member* should display arranged class members	`/classes/ CLASS01/ _actions/ manage- class- members`	• *Test case 13.2 should have been executed successfully* • Open the page in the web browser. If is in the page	Banner should be in default state. The tag input entry "John the Teacher Nguyễn (john_doe.teacher)" should be on the input field for teacher selection by default.	As expected; Demonstrated by Figure 14	Passed

			already, refresh the page.			
13.4	*Manage Class Member* page should facilitate updating class members	`/classes/ CLASS01/ _actions/ manage- class- members`	• *Test case 2.7* should have been executed successfully • Continues from the test steps of *test case 13.3* • Then, click on the input field for student selection, type 't', then click on the entry "Tommy the Student Hoàng (student)" from the dropdown menu (as demonstrated in figure 62)	*same as test case 13.2*	As expected; Demonstrated by Figure 16	Passed
	14. Class Action Page: Organise Weekly Schedule					
14.1	*Organise Weekly Schedule* page should be rendered correctly	`/classes/ CLASS01/ _actions/ organise- weekly- schedule`	• *Test cases 5.1* and *5.2* should have been executed successfully. • Continues from the test steps of *test case 12.2* to get to this page.	- Input fields for start date and end date should display placeholder text "From which date?" and "To which date (inclusively)?" respectively. - Table should have 2 rows, each for an existing timeslot - Table should be unresponsive by default	As expected; Demonstrated by Figure 17	Passed

14.2	*Organise Weekly Schedule* page should not allow submission when input dates are not chosen yet		• Continues from the test steps of *test case 14.1* to get to this page • In the page: Click on the "Submit" button	Banner has colour changed to red, with messages: - For "startDate": "Start date cannot be empty.". - For "endDate": "End date cannot be empty.".	As expected; Demonstrated by Figure 18	Passed
14.3	*Organise Weekly Schedule* page should inform user when it fails to establish connection with the server	`/classes/ CLASS01/ _actions/ organise- weekly- schedule`	• Continues from the test steps of *test case 14.2* • Shut down the server application • Select values for start date and end date • In the page: Click on the "Submit" button	Banner has colour changed to red, with message "Failed to establish connection with server. If the problem persists, please contact your admin."	- The page stuck at loading state, as demonstrated by figure 66. - The log statements written to console output suggested two `NetworkErro r` uncaught, as demonstrated by Figure 20.	Failed
14.4. 1	*Organise Weekly Schedule* page should facilitate class period selections when start date and end date are chosen	`/classes/ CLASS01/ _actions/ organise- weekly- schedule`	• Continues from the test steps of *test case 14.3* • Kick-start the server application • Re-load the page in the browser • Select values for start date and end date	Table for class period selection should be enabled for use.	As expected; Demonstrated by Figure 21	Passed

14.4.2			• Continues from the test steps of *test case 14.4.1* • Elect some class periods using the selection table, as shown in the screen capture of figure 69 • Click on the "Submit" button	Banner has colour changed to green, with messages "54 class periods created."	As expected; Demonstrated by Figure 22	Passed
15. Teachers pages						
15.1.	Info page for all teachers	`/teachers`	• *Test case 2.4* should has been executed successfully • Open this page in the web browser	Table should have 1 entry with: - "Teacher Name (Username)" of value to "John the Teacher Nguyễn (john_doe.teacher)". The display name text should link to the page "`/timetable?user-role=teacher&username=john_doe.teacher`" - "Date of Birth" of value "01/01/1980" - "Personal Email" of value "john_doe.teacher@domain.com" - "Phone Number" of value "+849041234568"	As expected; Demonstrated by Figure 23	Passed
15.2.	A teacher should be able to see their timetable		• Continues from test steps of *test case 15.1* • Follow the link of the teacher's display name	Same as *test case 12.3*, but: The banner title is "John the Teacher Nguyễn's Timetable"	As expected; Demonstrated by Figure 24	Passed
16. Students pages						

16.1.	Info page for all students	`/students`	• *Test case 2.7 should has been executed successfully* • Open this page in the web browser	Student table should have 1 entry with: - "Student Name (Username)" of value to "Tommy the Student Hoàng (student)". The display name text should link to the page "`/timetable?user-role=student&username=student`" - "Date of Birth" of value "21/12/1996" - "Personal Email" of value "tommy_learner_for_life@domain.com" - "Phone Number" of value "+84904123457"	As expected; Demonstrated by Figure 25	Passed
16.2.	A student should be able to see their timetable	`/timetable ?user-role =student &username =student`	• Continues from test steps of *test case 16.1* • Follow the link of the student's display name	Same as *test case 12.3*, but: - The banner title is "Tommy the Student Hoàng's Timetable"	As expected; Demonstrated by Figure 26	Passed
17. Class Periods pages						
17.1.1	Info page for an existing class period should be rendered correctly	`/classes /CLASS01 /2022-11-07/1/`	• *Test case 14.4.2 should have been executed successfully.*	There should be a link titled "Create New Homework" to "`/classes/CLASS01/2022-11-07/1/homework/_actions/add`" page.	As expected; Demonstrated by Figure 27	Passed
17.1.2			• When *test case 18.4* has been executed successfully.	There should be a link titled "Homework" to "`/classes/CLASS01/2022-11-07/1/homework/`" page.	As expected; Demonstrated by Figure 28	Passed
17.2	Info page for a non-existent class period	`/classes /CLASS01 /2023-11-07/1/`	• *Test case 14.4.2 should have been executed successfully*	Large red banner with message "Class 'CLASS01': There's no class periods at 07/11/2023 timeslot 1"	As expected; Demonstrated by Figure 29	Passed

18. Class Period Homework pages
18.1	Page for create homework for a class period should be rendered correctly	/classes /CLASS01 /2022-11- 07/1 /homework /_actions/ad d	• Continues from the test steps of *test case 17.1.1*, follow the link to get to the page under test	By default, the "Submit" button should be unclickable	As expected; Demonstrated by Figure 30	Passed
18.2	Page for create class homework should validate against question not having any options		• Continues from the test steps of *test case 18.1* • Type the first question, then hit *Enter*, as demonstrated by figure 78 • Click "Submit" button	Banner has colour turned to red, with a message for "Question 1" displayed as "No answers defined yet."	As expected; Demonstrated by figure 78	Passed
18.3	Page for create class homework should validate against question not having any options selected as the right one	/classes /CLASS01 /2022-11- 07/1 /homework /_actions/ad d	• Continues from the test steps of *test case 18.2* • Input some question answers for the defined questions • Click "Submit" button	Banner colour is red, with a message for "Question 1" displayed as "No answers elected as the right one yet."	As expected; Demonstrated by Figure 32	Passed
18.4	Page for create class homework should be functional		• Continues from the test steps of *test case 18.3* • Select an option as the question's right answer • Click "Submit" button	- Banner has colour turned to green with message "1 question(s) added for homework." - "Submit" button should disappear	As expected; Demonstrated by Figure 33	Passed

18.5	Page for viewing added homework should display correctly	`/classes /CLASS01 /2022-11-07/1 /homework`	• *Test case 18.4 should have been executed successfully*	The added question options are displayed correctly as defined	As expected; Demonstrated by Figure 34	Passed
19. Miscellaneous pages						
19.1	Page generated by Swagger UI should be accessible and up-to-date	`/swagger/ind ex.html`	• Open the page in the web browser	The page should be display all the API routes defined by the server application	As expected; Demonstrated by Figure 35	Passed

Screenshots
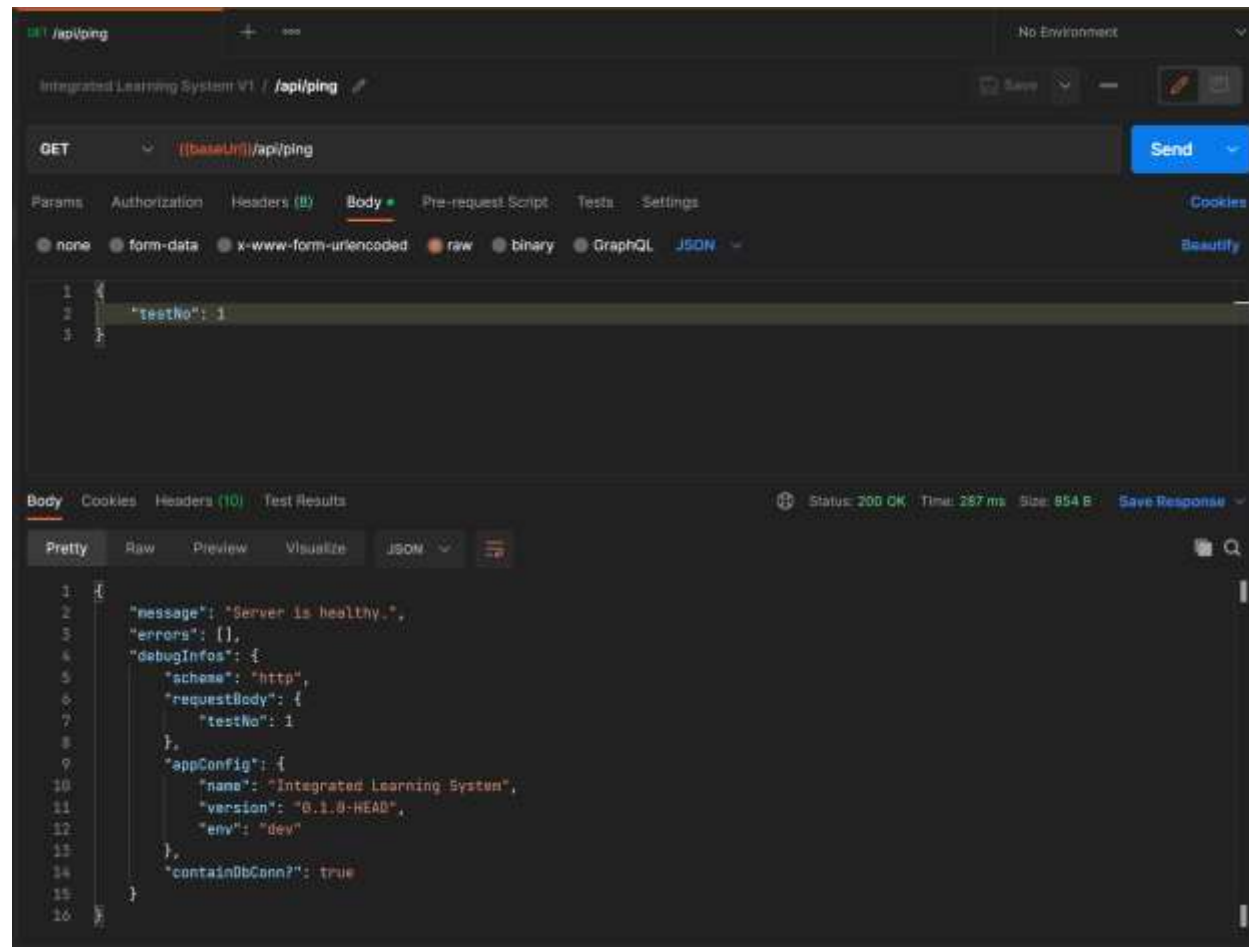
Figure 1. Request and response of test case 1.1
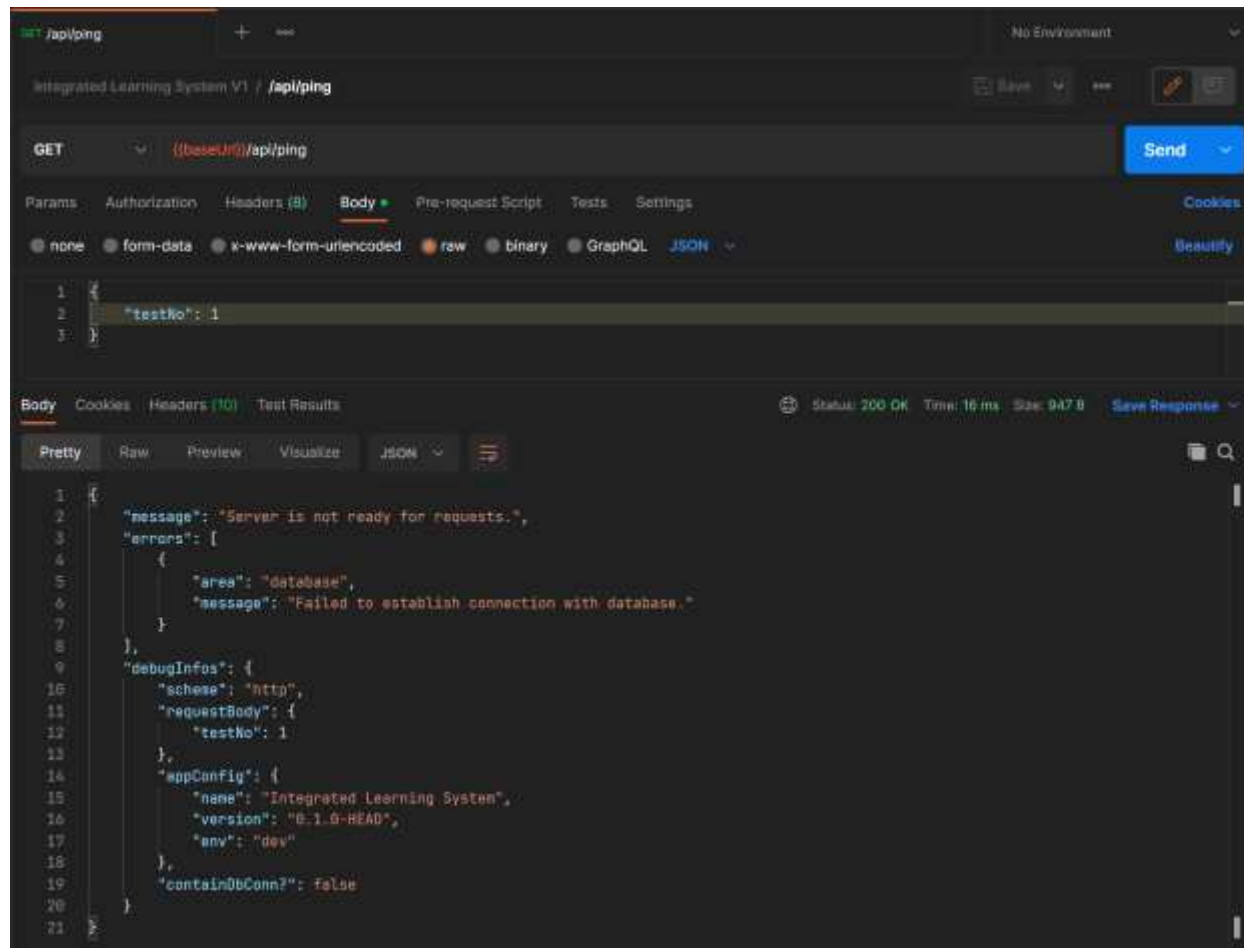
Figure 2. Request and response of test case 1.2

```
1    {
2      "classTeacher": {
3        "username": "john_doe.teacher",
4        "firstName": "John the Teacher",
5        "lastName": "Nguyễn",
6        "fullName": "John the Teacher Nguyễn",
7        "displayName": "John the Teacher Nguyễn (john_doe.teacher)",
8        "timetable": [
9          { "timeslotNumber": 1, "schoolDate": "2022-10-03", "className": "CLASS01", "courseCode": "LEARNING101", "courseName": "Learning 101" },
10         { "timeslotNumber": 2, "schoolDate": "2022-10-03", "className": "CLASS01", "courseCode": "LEARNING101", "courseName": "Learning 101" },
11         { "timeslotNumber": 2, "schoolDate": "2022-10-04", "className": "CLASS01", "courseCode": "LEARNING101", "courseName": "Learning 101" }
12       ]
13     },
14     "classStudents": [
15       {
16         "username": "student",
17         "firstName": "Tommy the Student",
18         "lastName": "Hoàng",
19         "fullName": "Tommy the Student Hoàng",
20         "displayName": "Tommy the Student Hoàng (student)",
21         "timetable": [
22           { "timeslotNumber": 1, "schoolDate": "2022-10-03", "className": "CLASS01", "courseCode": "LEARNING101", "courseName": "Learning 101" },
23           { "timeslotNumber": 2, "schoolDate": "2022-10-03", "className": "CLASS01", "courseCode": "LEARNING101", "courseName": "Learning 101" },
24           { "timeslotNumber": 2, "schoolDate": "2022-10-04", "className": "CLASS01", "courseCode": "LEARNING101", "courseName": "Learning 101" }
25         ]
26       }
27     ]
28   }
```

Figure 3. Actual response body of test case 9.5

Figure 4. Webpage displayed of test case 12.1
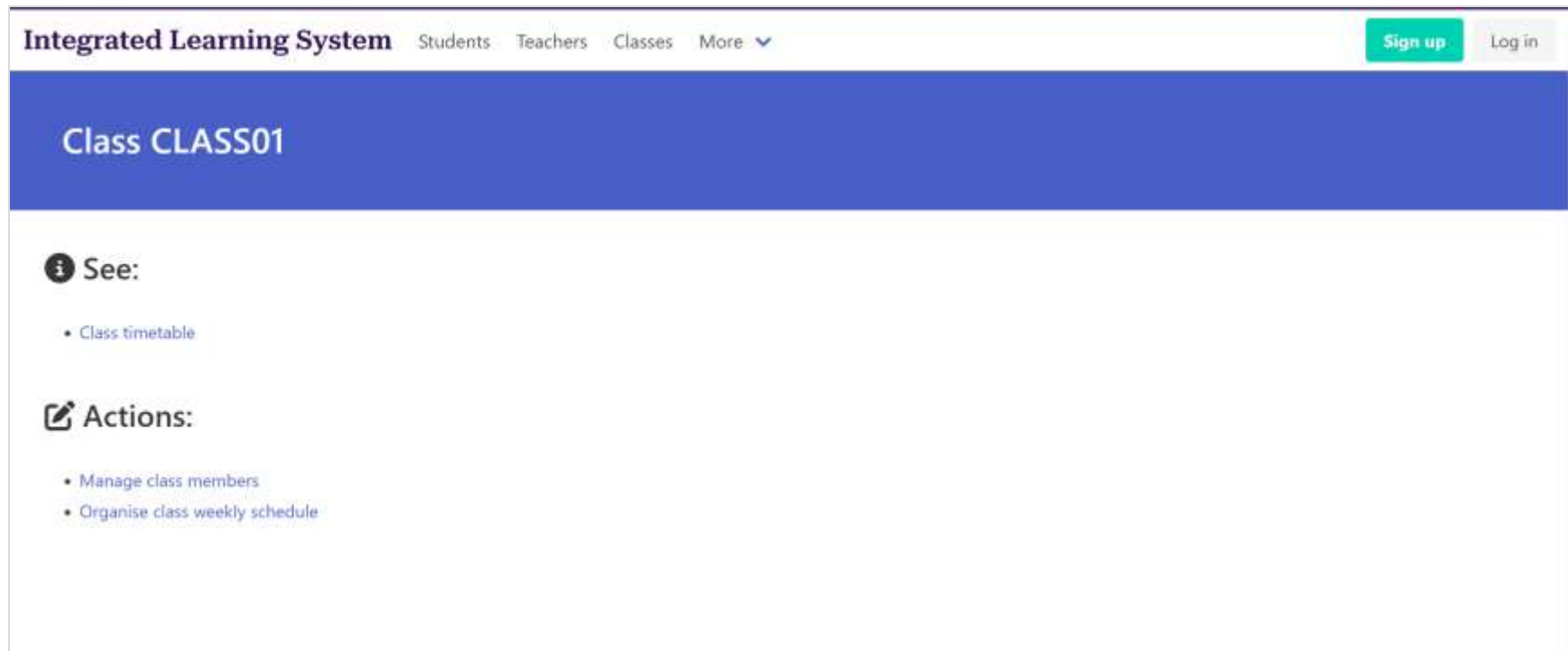
Figure 5. Webpage displayed of test case 12.2

Figure 6. Demonstrations for test case 12.3, webpage rendered for week 41 of 2022

Figure 7. Demonstrations for test case 12.3, webpage rendered for week 42 of 2022

Figure 8. Demonstrations for test case 12.3, webpage rendered for week 44 of 2022

Figure 9. Demonstrations for test case 12.3, webpage rendered for week 48 of 2022

Figure 10. Demonstrations for test case 12.3, webpage rendered for week 49 of 2022

Figure 11. Demonstrations for test case 12.3, webpage rendered for week 50 of 2022

Figure 12. Webpage displayed of test case 13.1

Figure 13. Webpage displayed of test case 13.2
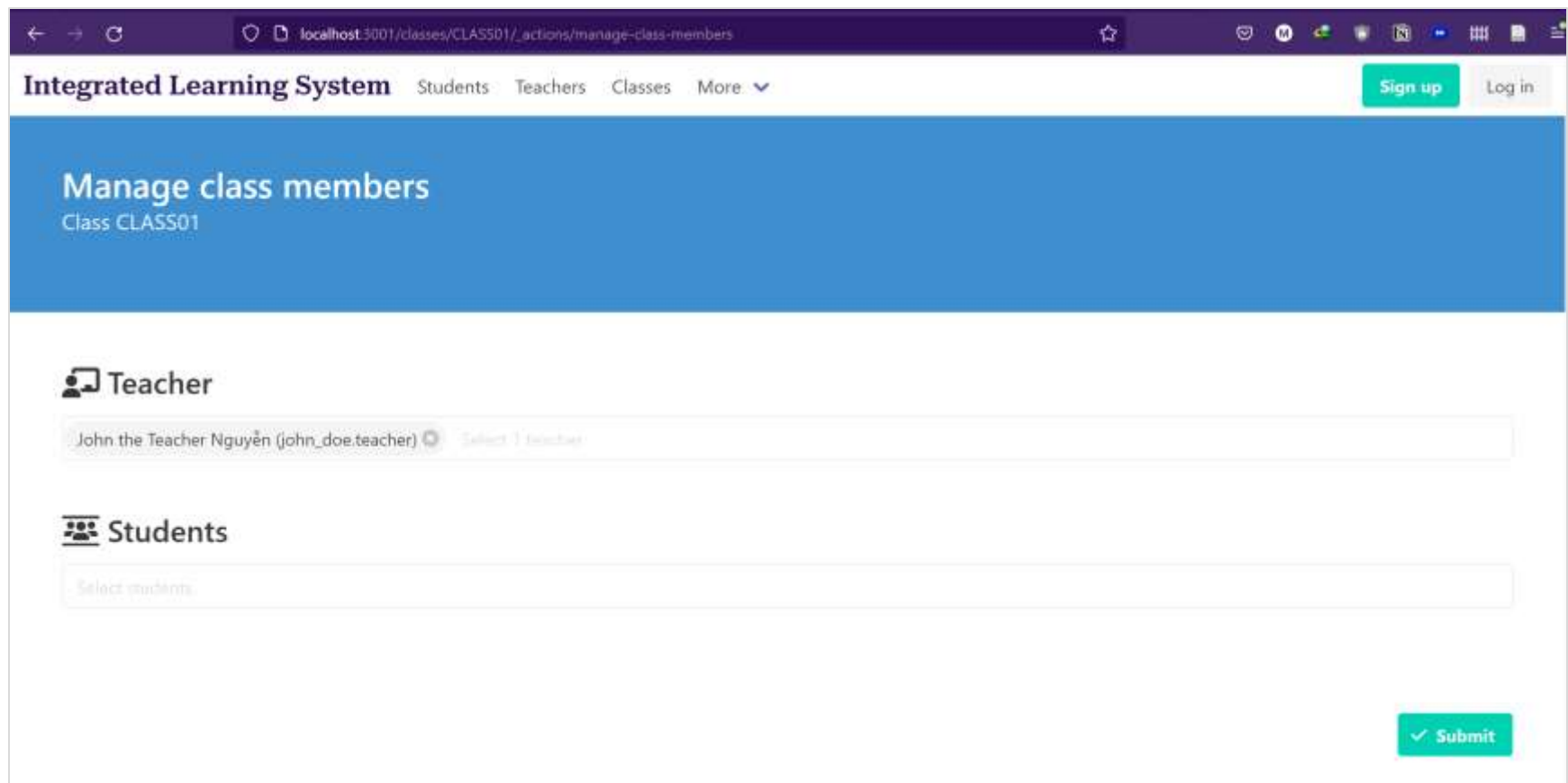
Figure 14. Webpage displayed of test case 13.3

Figure 15. Webpage screen capture demonstrating test steps of test case 13.4

Figure 16. Webpage displayed for test case 13.4

Figure 17. Test case 14.1 demonstration

Figure 18. Test case 14.2 demonstration

Figure 19. Test case 14.3 webpage demonstration

Figure 20. Test case 14.3 console output demonstration

Figure 21. Test case 14.4.1 demonstration

Figure 22. Test case 14.4.2 result demonstration

Figure 23. Test case 15.1 webpage demonstration

Figure 24. Test case 15.2 demonstrations, webpage rendered for week 46 of 2022

Figure 25. Test case 16.1 webpage demonstration

Figure 26. Test case 16.2 demonstrations, webpage rendered for week 43 of 2022
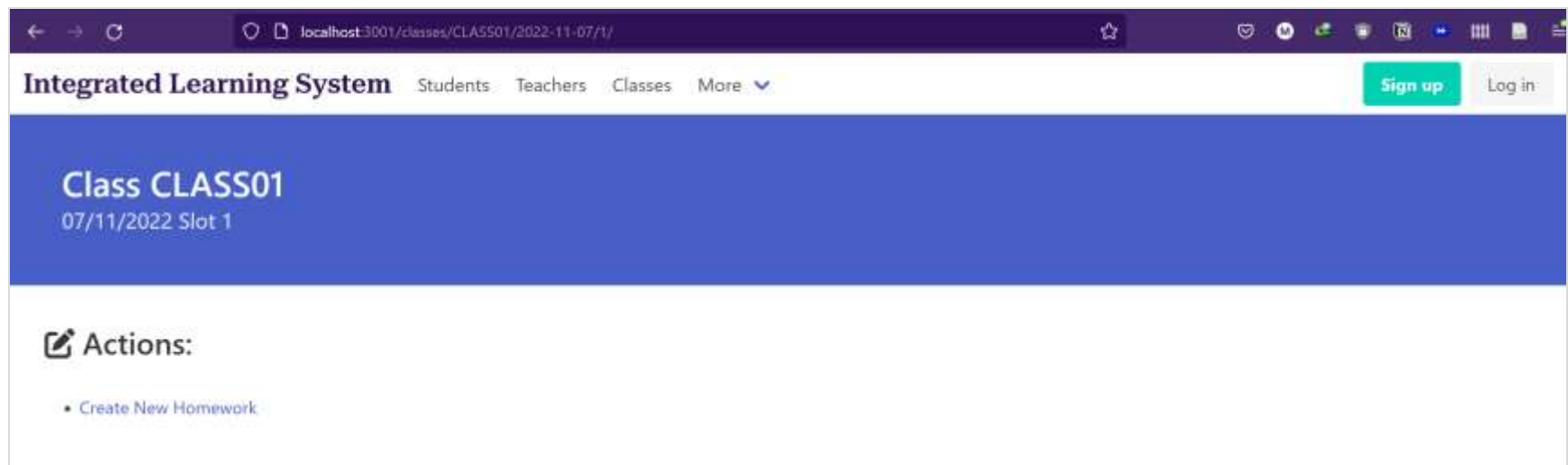
Figure 27. Webpage demonstration for test case 17.1.1
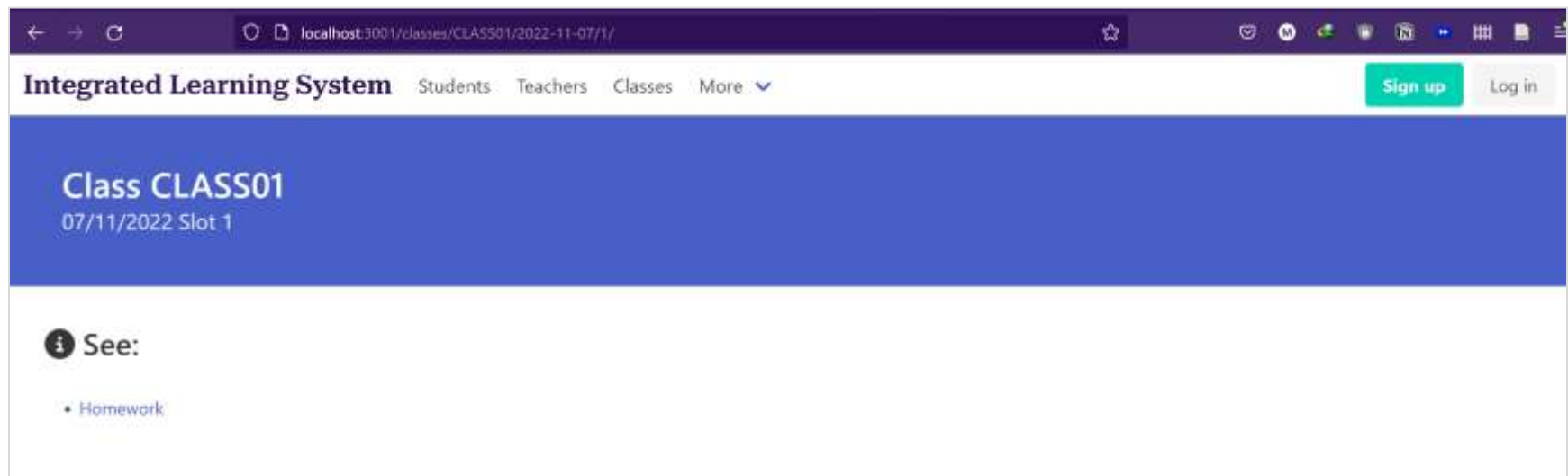
Figure 28. Webpage demonstration for test case 17.1.2
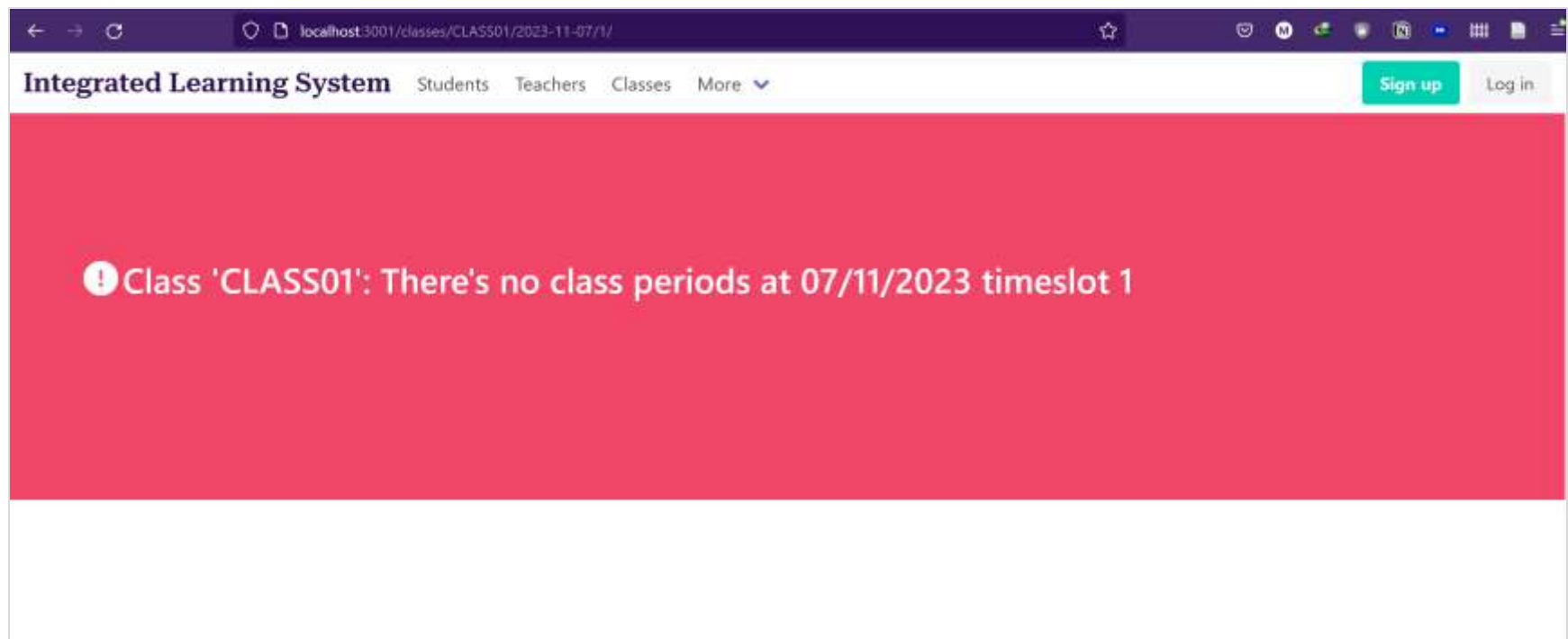
Figure 29. Webpage demonstration for test case 17.2

Figure 30. Webpage demonstration for test case 18.1

Figure 31. Test case 18.2 webpage demonstration
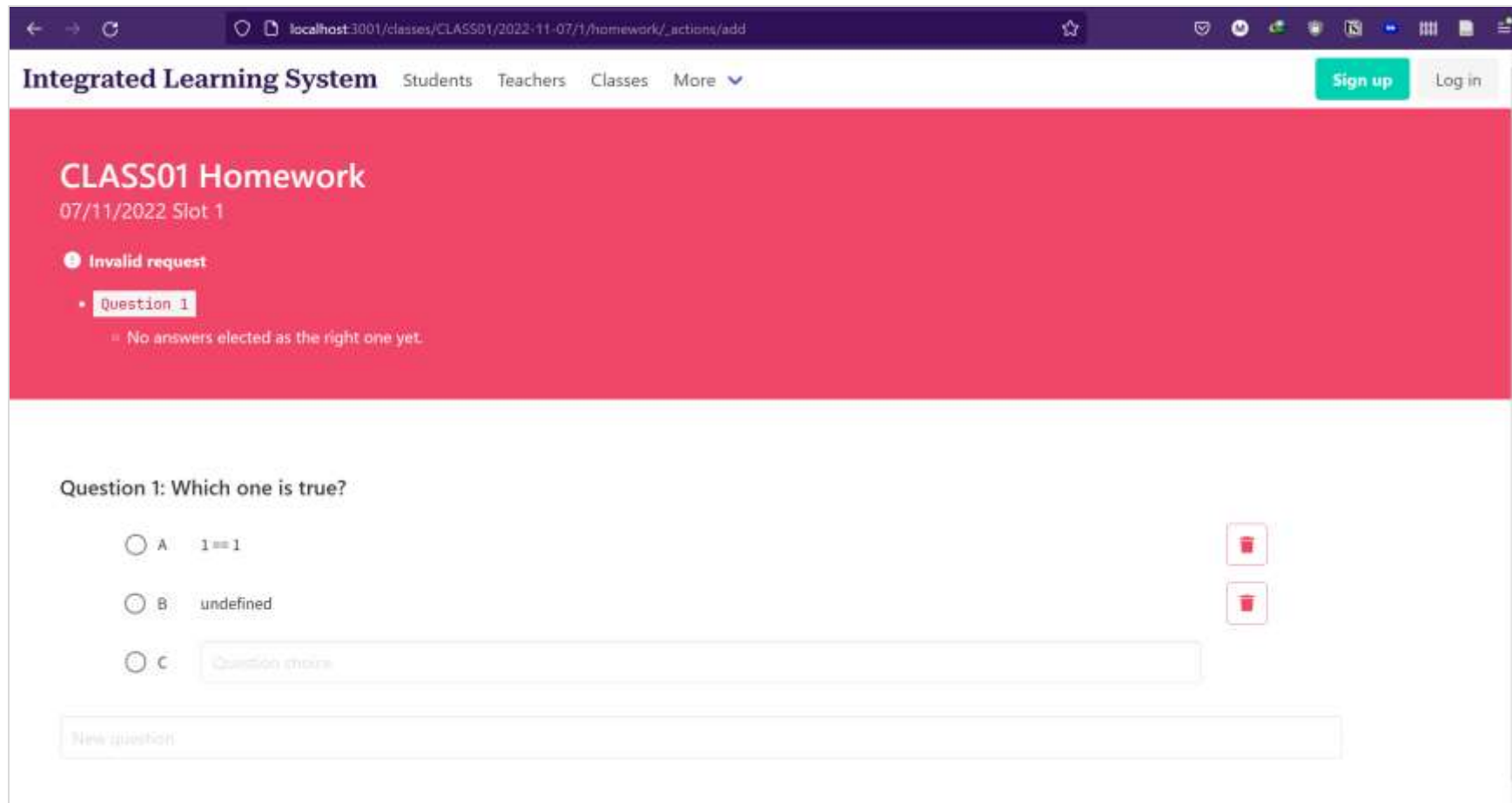
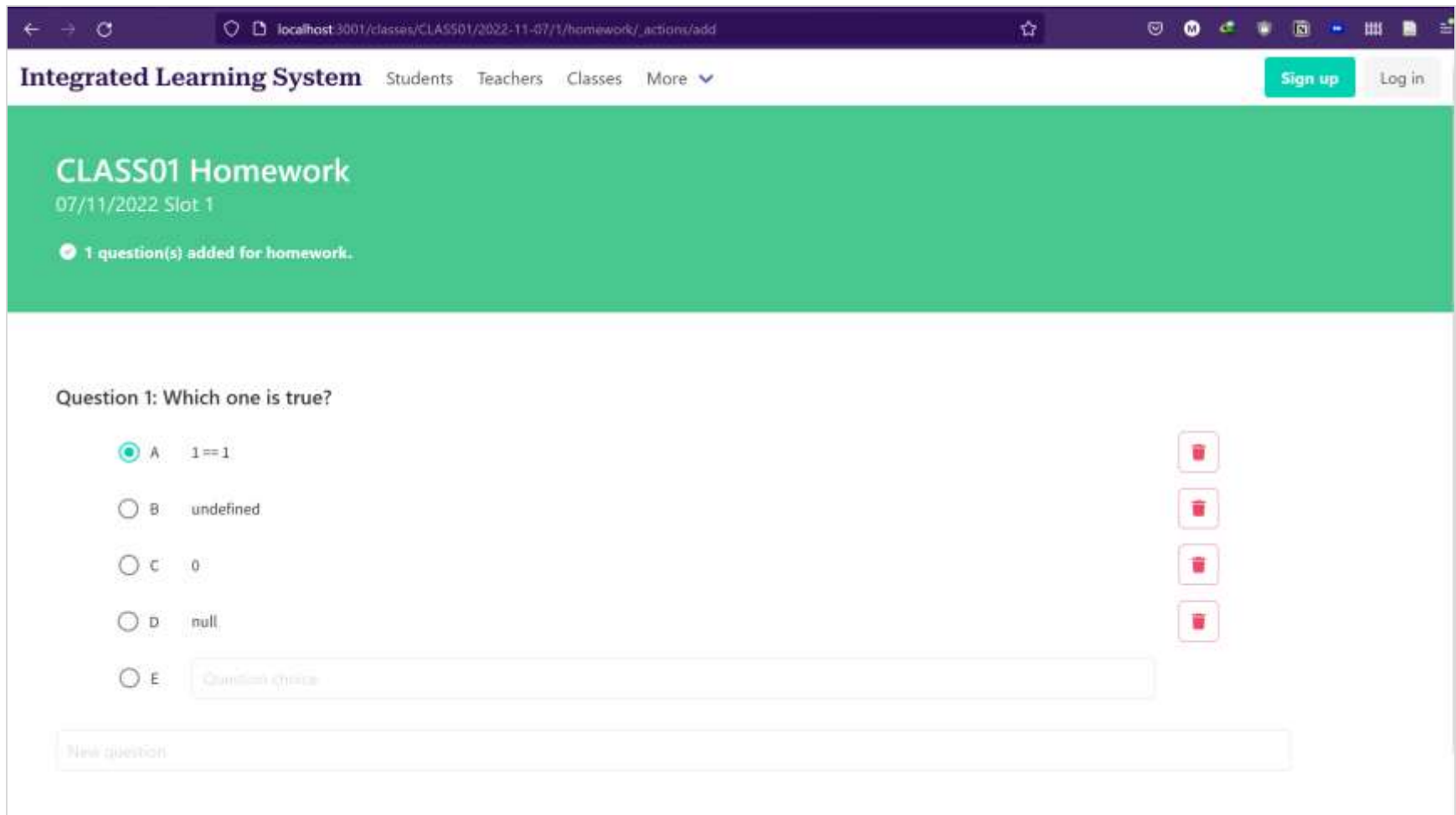*Figure 32. Test case 18.3 webpage demonstration*

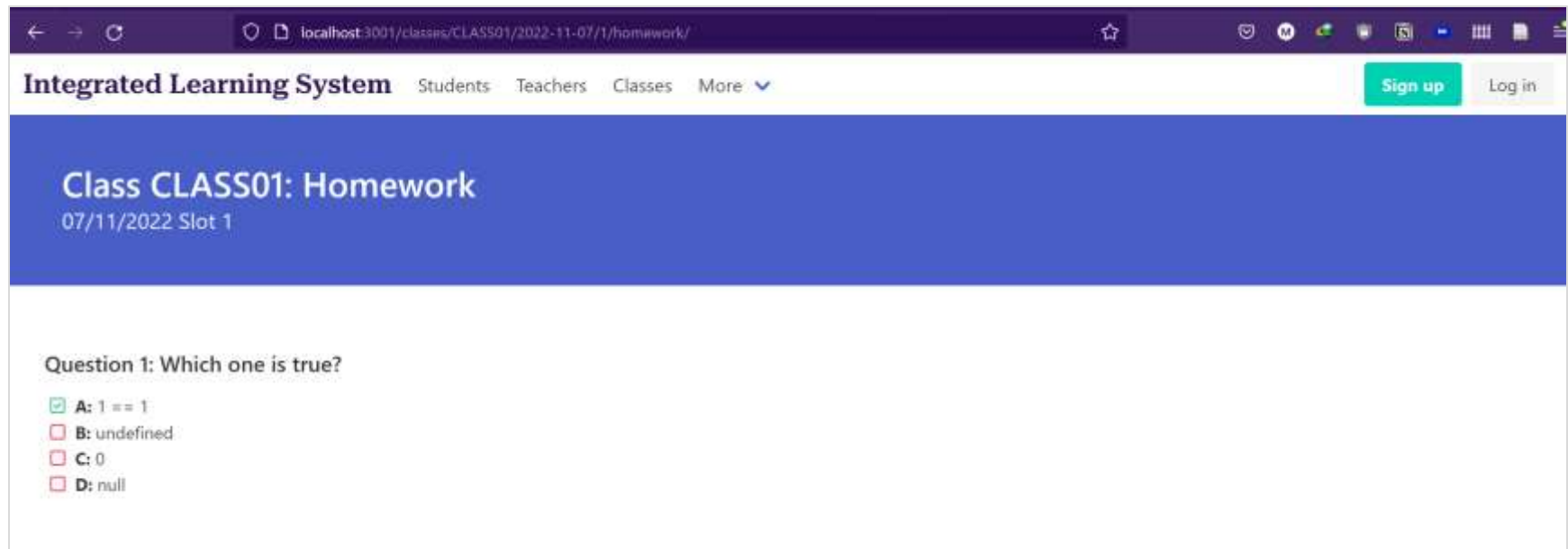*Figure 33. Test case 18.4 webpage demonstration*

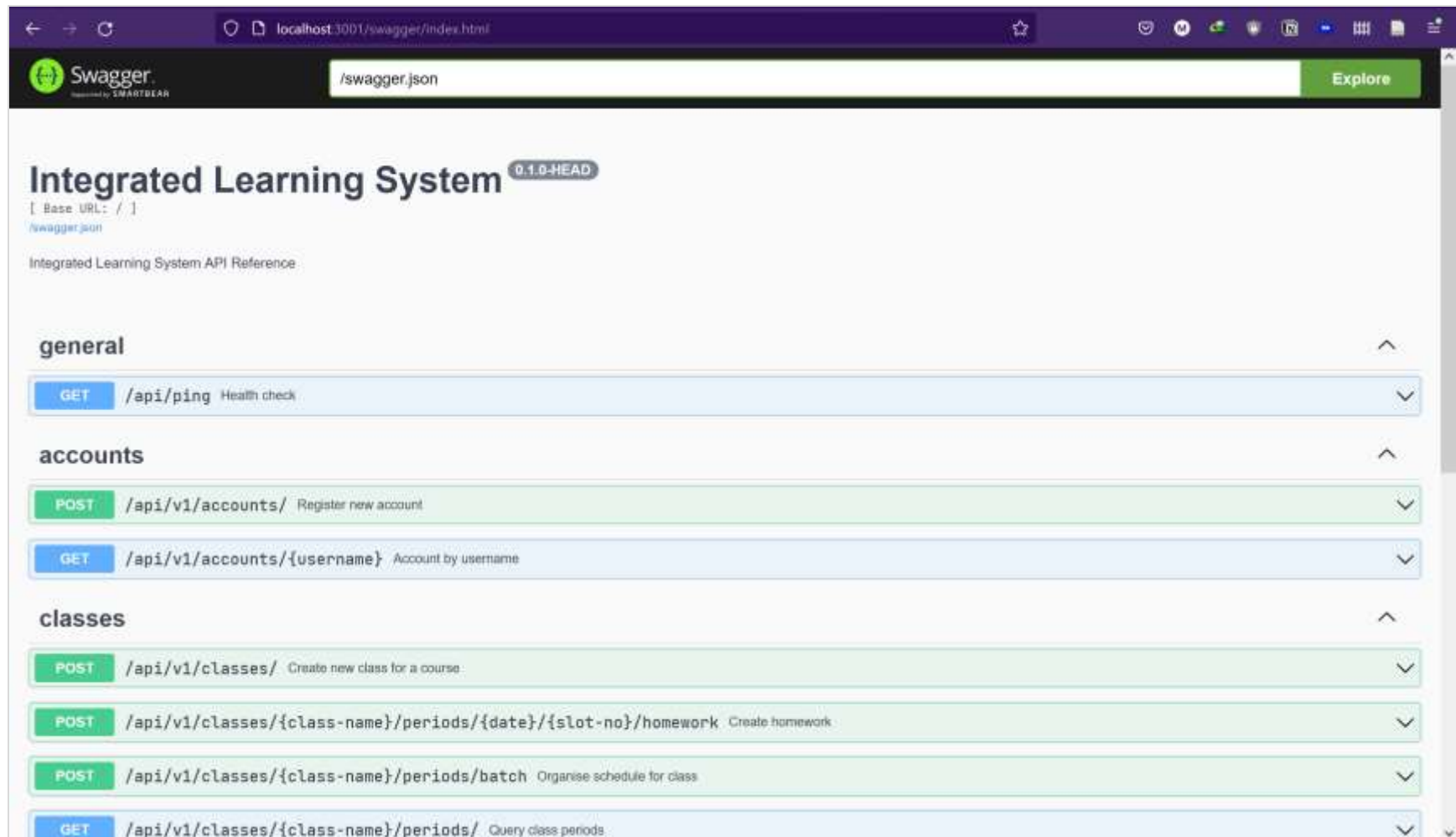*Figure 34. Test case 18.5 webpage demonstration*

Figure 35. A screen capture of the webpage demonstration for test case 19.1