

Report: Database Schema Definition SQL Queries

TABLE: COLLEGES

```
CREATE TABLE colleges (  
    id SERIAL PRIMARY KEY,  
    name VARCHAR(100) NOT NULL UNIQUE  
);
```

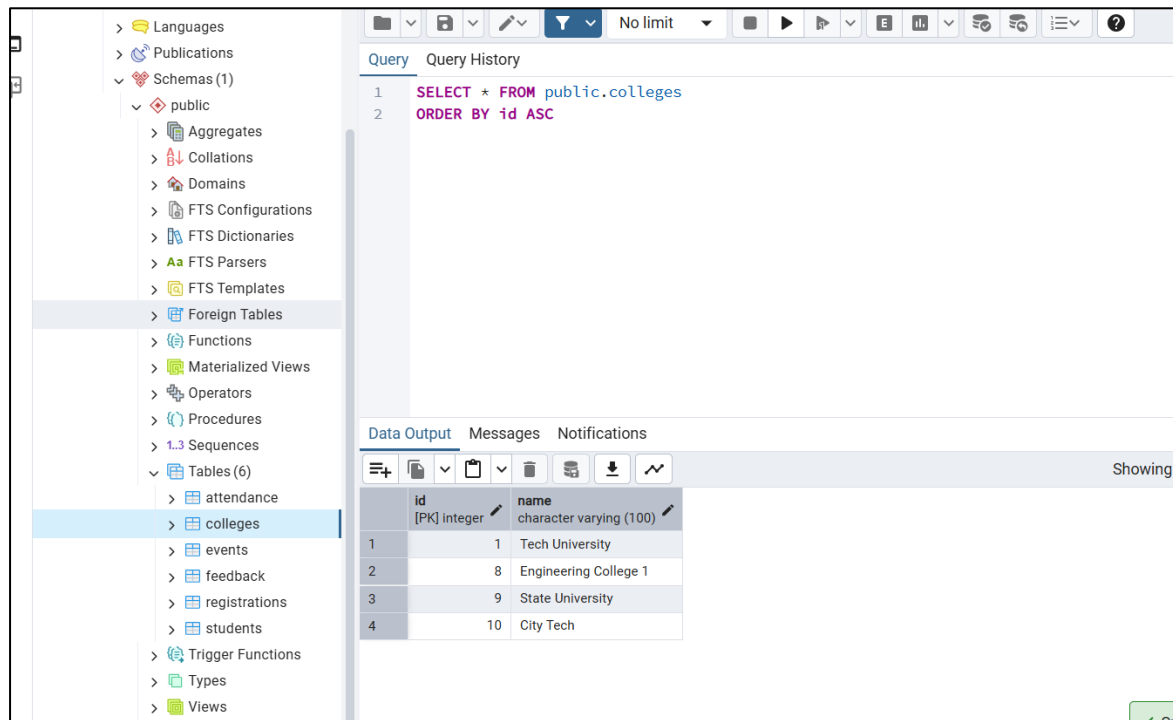


Fig.1: Table colleges

TABLE: STUDENTS

```
CREATE TABLE students (  
    id SERIAL PRIMARY KEY,  
    name VARCHAR(100) NOT NULL,  
    college_id INTEGER NOT NULL,  
    FOREIGN KEY (college_id) REFERENCES colleges(id) ON DELETE CASCADE  
);
```

Publications

Schemas (1)

public

Aggregates

Collations

Domains

FTS Configurations

FTS Dictionaries

FTS Parsers

FTS Templates

Foreign Tables

Functions

Materialized Views

Operators

Procedures

1.3 Sequences

Tables (6)

attendance

colleges

events

feedback

registrations

students

Trigger Functions

Types

Views

Subscriptions

Query

Query History

1 SELECT * FROM public.students

2 ORDER BY id ASC

Data Output

Messages

Notifications

	id [PK] integer	name character varying (100)	college_id integer
1	1	Alice Johnson	1
2	2	Bob Smith	1
3	11	Charlie Davis	8
4	12	Dana Lee	8
5	13	Eliot Grant	9
6	14	Fiona Wells	10

Fig.2: Table students

TABLE: EVENTS

CREATE TABLE events (

id **SERIAL PRIMARY KEY**,

name **VARCHAR**(100) NOT NULL,

type **VARCHAR**(50) NOT NULL,

date **DATE** NOT NULL,

college_id **INTEGER** NOT NULL,

FOREIGN KEY (college_id) **REFERENCES** colleges(id) **ON DELETE CASCADE**

);

The screenshot shows a database management interface. On the left, a tree view lists database objects under the 'public' schema, including 'Tables (6)' with 'events' selected. The main area displays a query: `SELECT * FROM public.events ORDER BY id ASC`. Below the query, the 'Data Output' tab shows the table structure and data rows.

	id [PK] integer	name character varying (100)	type character varying (50)	date date	college_id integer
1	1	Hackathon 2025	Workshop	2025-09-15	1
2	5	Tech Fest 2025	Fest	2025-09-20	1
3	6	AI Seminar	Seminar	2025-10-01	8
4	7	Coding Contest	Hackathon	2025-09-30	9

Fig.3: Table events

TABLE: REGISTRATIONS

```
CREATE TABLE registrations (
  id SERIAL PRIMARY KEY,
  student_id INTEGER NOT NULL,
  event_id INTEGER NOT NULL,
  UNIQUE(student_id, event_id),
  FOREIGN KEY (student_id) REFERENCES students(id) ON DELETE CASCADE,
  FOREIGN KEY (event_id) REFERENCES events(id) ON DELETE CASCADE
);
```

The screenshot shows a database management interface. On the left, a tree view displays the database structure, including Schemas (1), public, and various database objects like Aggregates, Collations, Domains, FTS Configurations, FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, Tables (6), Trigger Functions, Types, and Views. The 'registrations' table is highlighted under the 'Tables (6)' category.

The main pane shows a query editor with the following SQL query:

```
1 SELECT * FROM public.registrations
2 ORDER BY id ASC
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format. The table has four columns: 'id' (integer, primary key), 'student_id' (integer), 'event_id' (integer), and an unlabeled column. The results are as follows:

	id [PK] integer	student_id integer	event_id integer	
1	1	1	1	
2	3	2	1	
3	4	11	6	
4	5	12	6	
5	6	13	7	
6	7	14	7	
7	9	2	5	

Fig.4: Table registrations

TABLE: ATTENDANCE

CREATE TABLE attendance (

id SERIAL PRIMARY KEY,

student_id INTEGER NOT NULL,

event_id INTEGER NOT NULL,

status BOOLEAN NOT NULL,

UNIQUE(student_id, event_id),

FOREIGN KEY (student_id) REFERENCES students(id) ON DELETE CASCADE,

FOREIGN KEY (event_id) REFERENCES events(id) ON DELETE CASCADE

);

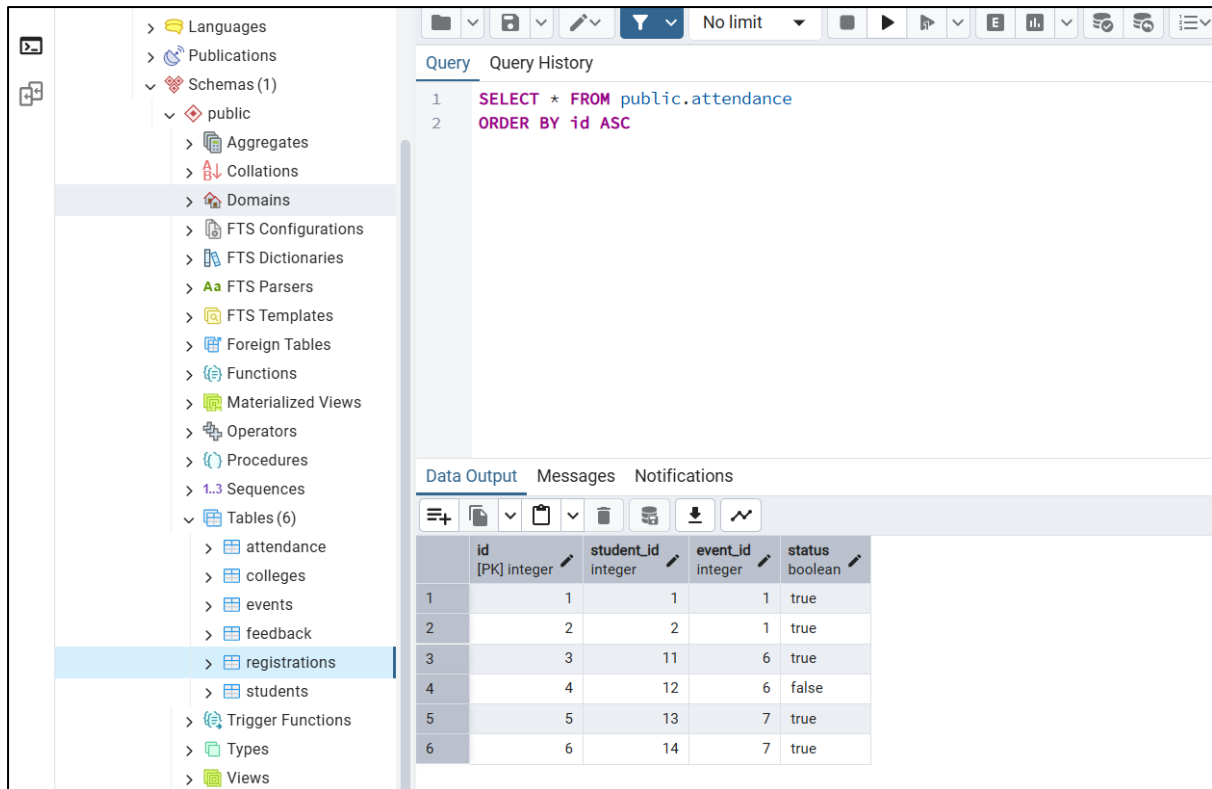


Fig.5: Table attendance

TABLE: FEEDBACK

CREATE TABLE feedback (

id **SERIAL PRIMARY KEY**,

student_id **INTEGER NOT NULL**,

event_id **INTEGER NOT NULL**,

rating **INTEGER NOT NULL CHECK** (rating BETWEEN 1 AND 5),

UNIQUE(student_id, event_id),

FOREIGN KEY (student_id) **REFERENCES** students(id) **ON DELETE CASCADE**,

FOREIGN KEY (event_id) **REFERENCES** events(id) **ON DELETE CASCADE**

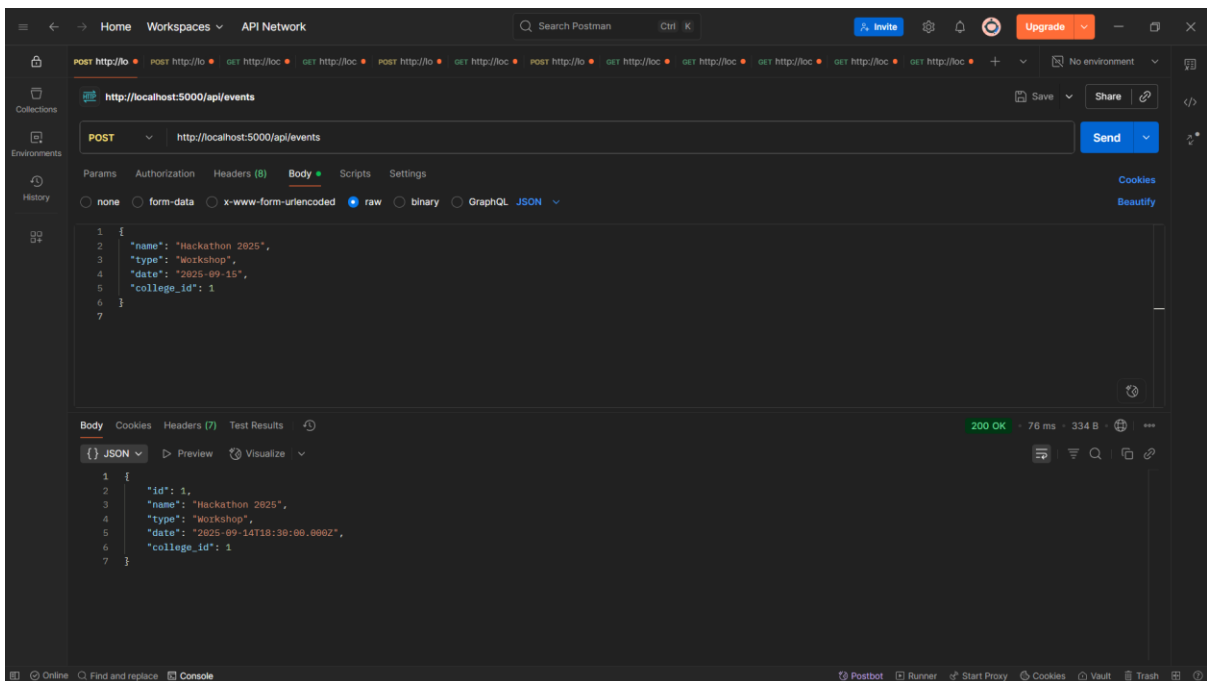
);

The screenshot shows a database management interface. On the left, a tree view displays the database structure, including Schemas, Tables, and Views. The 'public' schema is expanded, showing various database objects. The 'feedback' table is selected. In the center, a SQL query is entered: `SELECT * FROM public.feedback ORDER BY id ASC`. On the right, the 'Data Output' tab displays the results of the query in a table format.

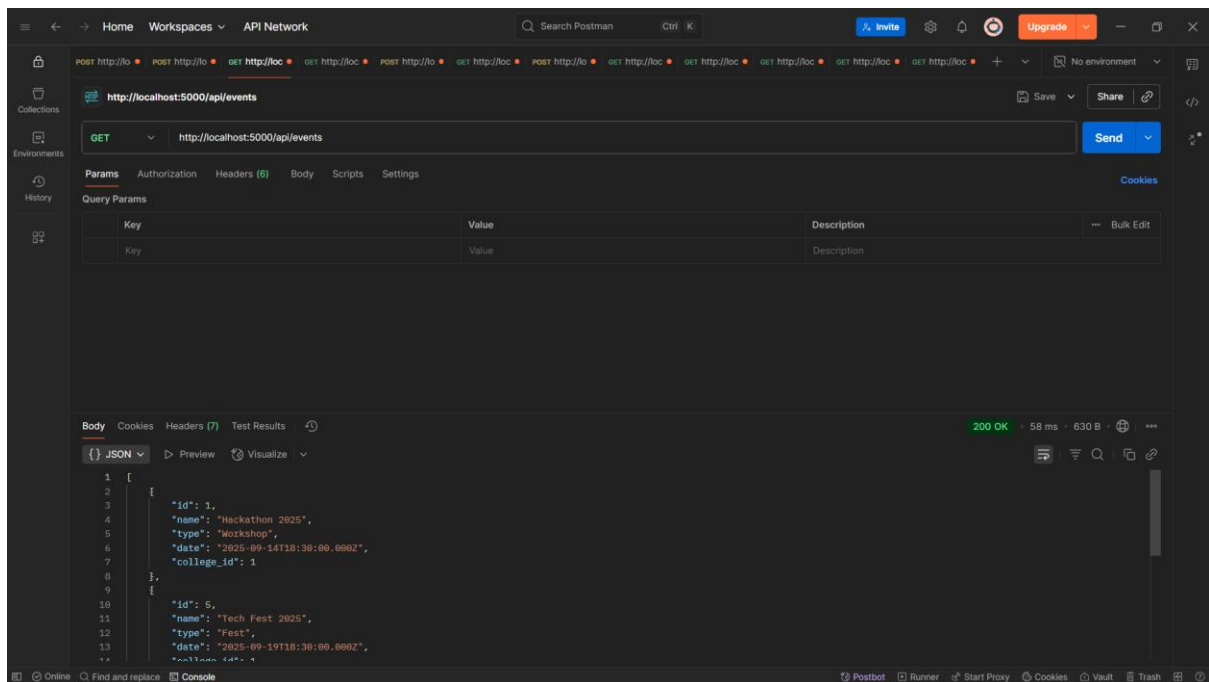
	id [PK] integer	student_id integer	event_id integer	rating integer
1	1	1	1	4
2	2	2	1	5
3	3	11	6	4
4	4	13	7	4
5	5	14	7	5

Fig.6: Table feedback

I have used **POSTMAN** for my API endpoints testing, below are the screenshots of the Requests and Responses. I have included the API calls in the design document, I'll be including the JSON responses under each image respectively.

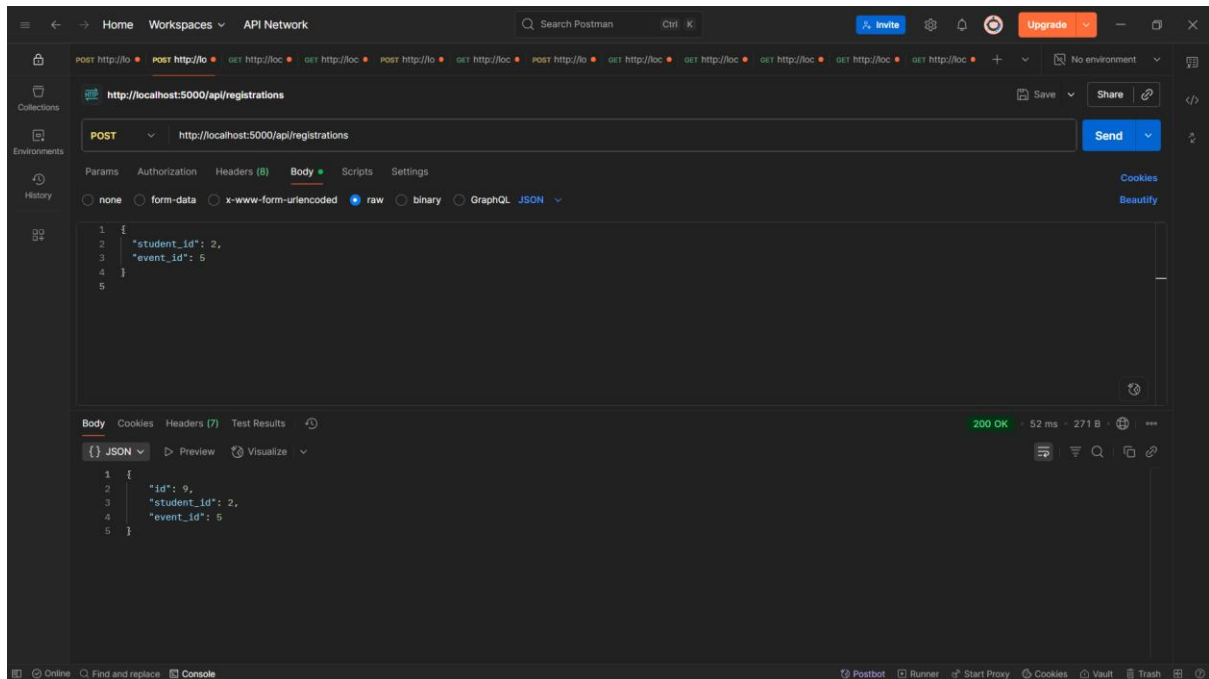


```
{
  "id": 1,
  "name": "Hackathon 2025",
  "type": "Workshop",
  "date": "2025-09-14T18:30:00.000Z",
  "college_id": 1
}
```

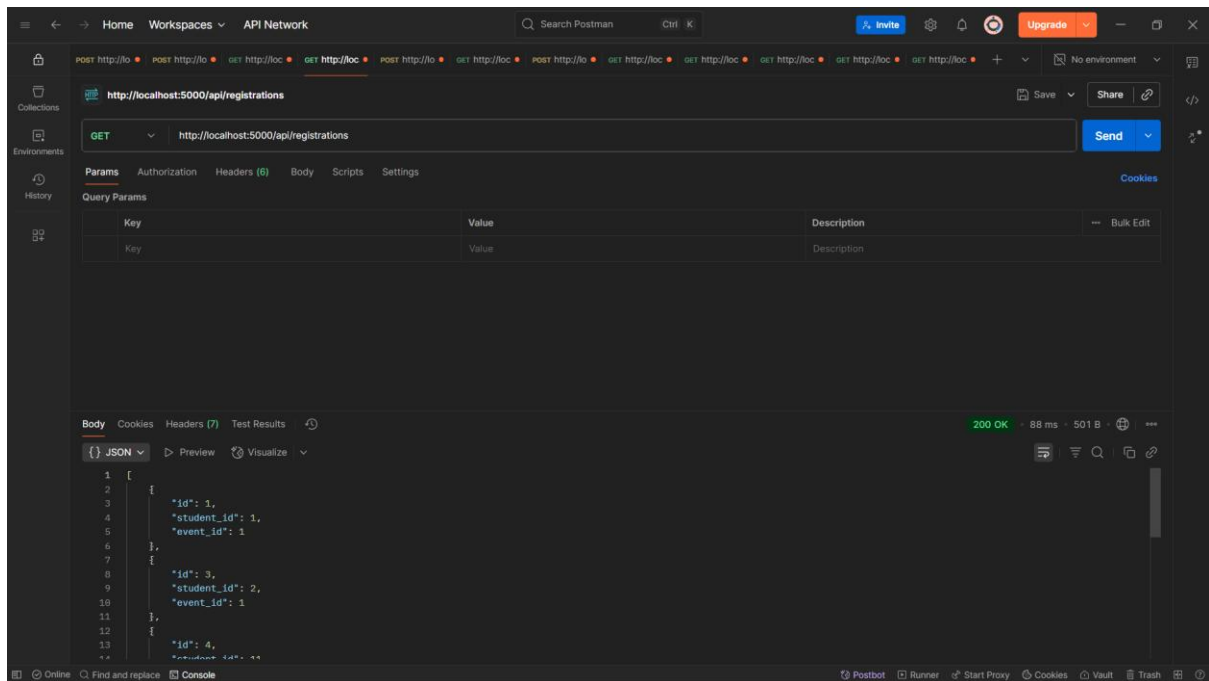


```
[
  {
    "id": 1,
    "name": "Hackathon 2025",
    "type": "Workshop",
    "date": "2025-09-14T18:30:00.000Z",
    "college_id": 1
  },
  {
    "id": 5,
    "name": "Tech Fest 2025",
    "type": "Fest",
```

```
"date": "2025-09-19T18:30:00.000Z",
"college_id": 1
},
{
  "id": 6,
  "name": "AI Seminar",
  "type": "Seminar",
  "date": "2025-09-30T18:30:00.000Z",
  "college_id": 8
},
{
  "id": 7,
  "name": "Coding Contest",
  "type": "Hackathon",
  "date": "2025-09-29T18:30:00.000Z",
  "college_id": 9
}
]
```

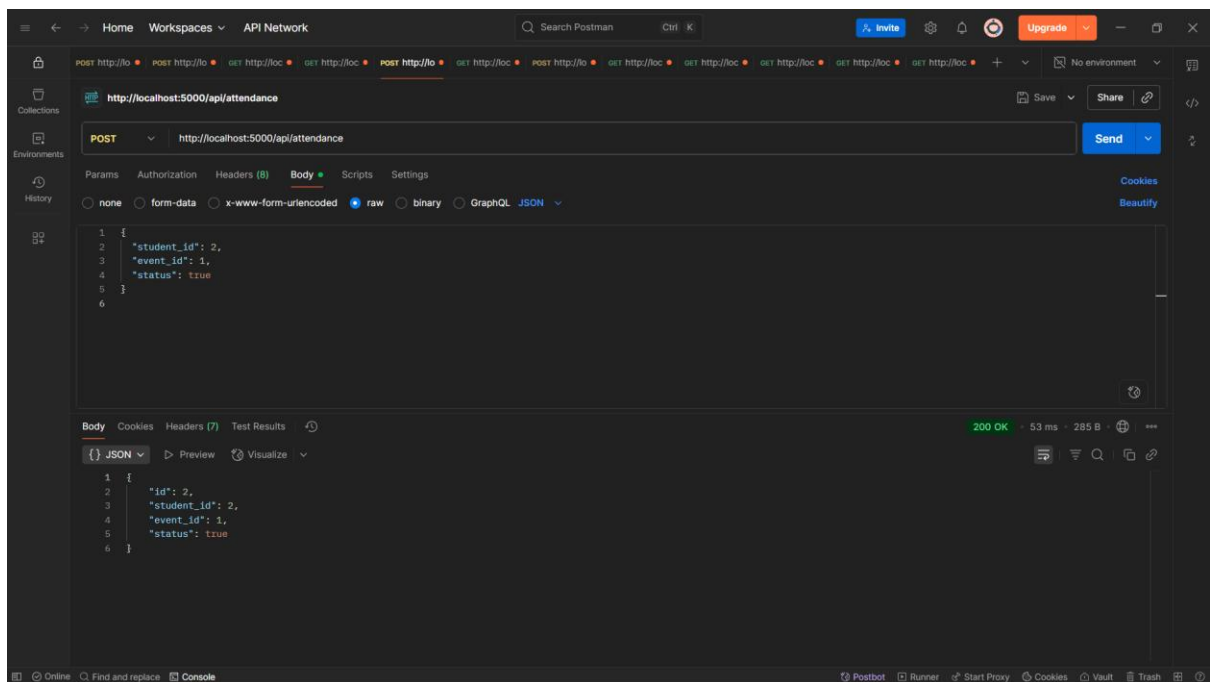



```
{
  "id": 9,
  "student_id": 2,
  "event_id": 5
}
```

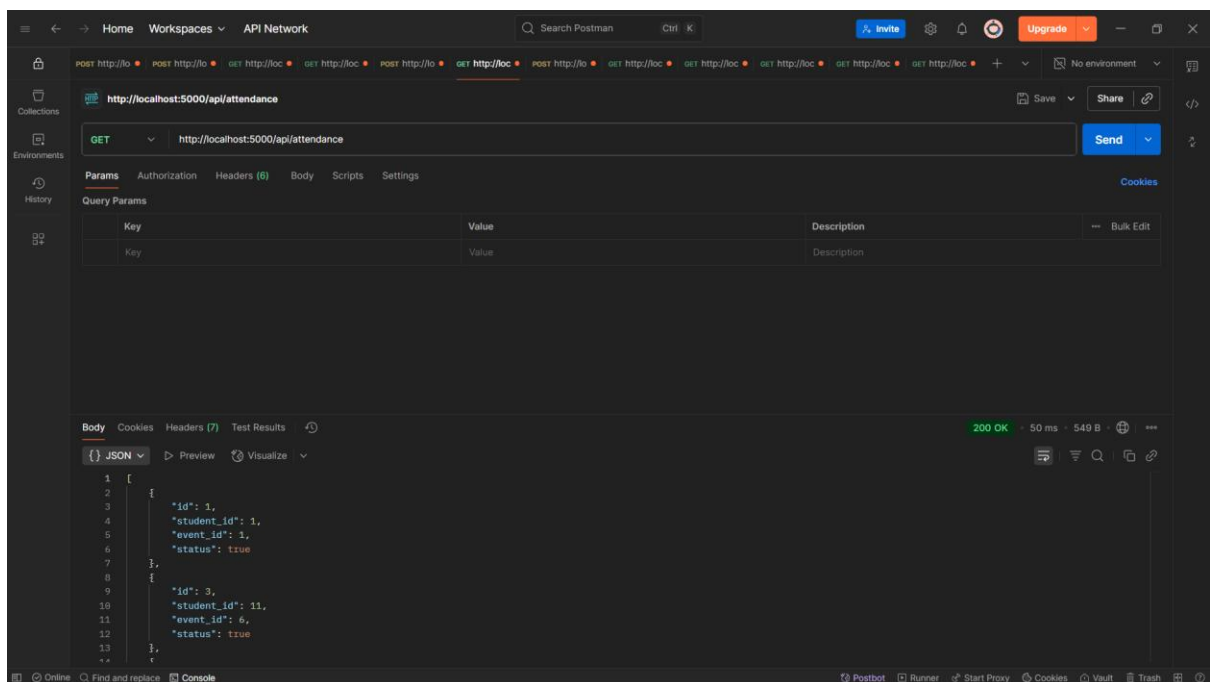


```
[
  {
    "id": 1,
    "student_id": 1,
    "event_id": 1
  },
  {
    "id": 3,
    "student_id": 2,
    "event_id": 1
  },
  {
    "id": 4,
    "student_id": 11,
    "event_id": 1
  }
]
```

```
    "event_id": 6
  },
  {
    "id": 5,
    "student_id": 12,
    "event_id": 6
  },
  {
    "id": 6,
    "student_id": 13,
    "event_id": 7
  },
  {
    "id": 7,
    "student_id": 14,
    "event_id": 7
  },
  {
    "id": 9,
    "student_id": 2,
    "event_id": 5
  }
]
```



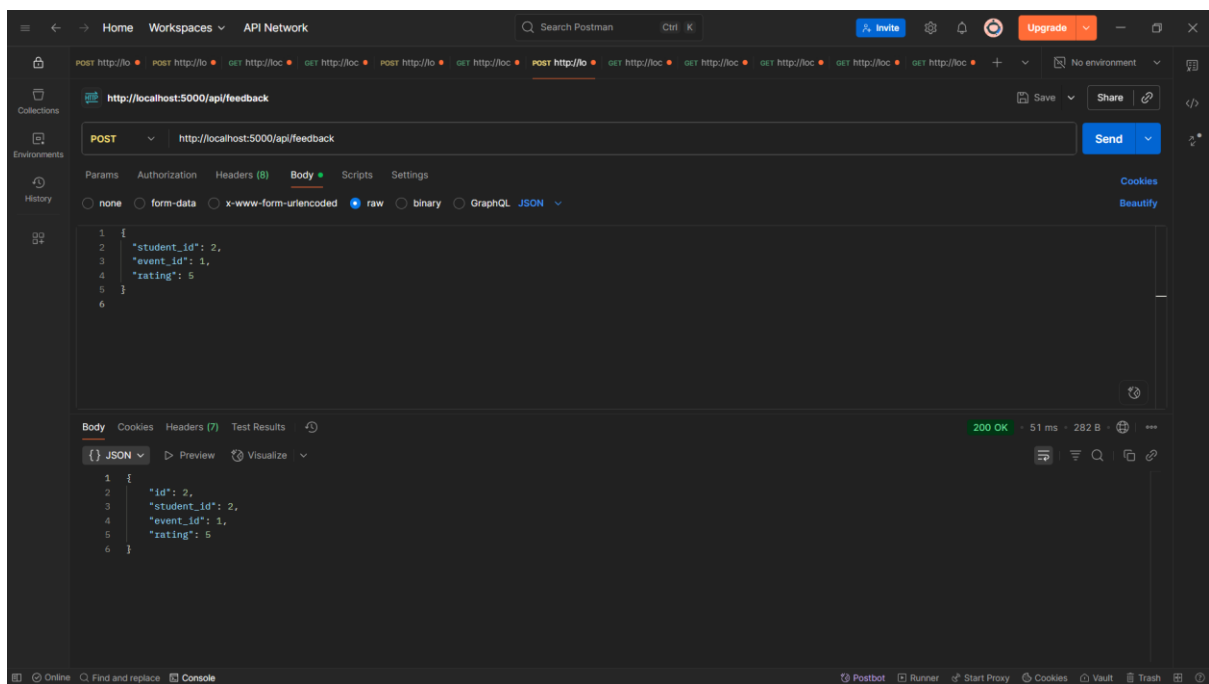
```
{
  "id": 2,
  "student_id": 2,
  "event_id": 1,
  "status": true
}
```



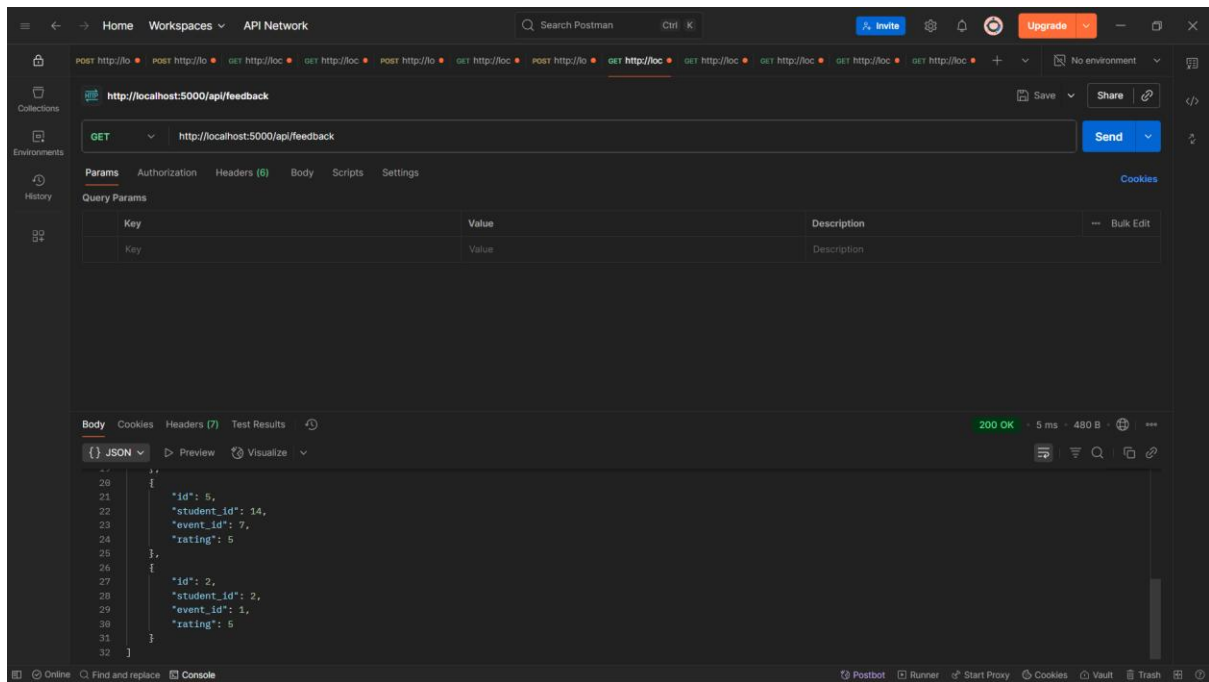
```
[
```

```
{
  "id": 1,
  "student_id": 1,
  "event_id": 1,
  "status": true
},
{
  "id": 3,
  "student_id": 11,
  "event_id": 6,
  "status": true
},
{
  "id": 4,
  "student_id": 12,
  "event_id": 6,
  "status": false
},
{
  "id": 5,
  "student_id": 13,
  "event_id": 7,
  "status": true
},
{
  "id": 6,
  "student_id": 14,
  "event_id": 7,
  "status": true
},
{
  "id": 2,
```

```
]
{
  "status": true,
  "event_id": 1,
  "student_id": 2,
}
```



```
{
  "id": 2,
  "student_id": 2,
  "event_id": 1,
  "rating": 5
}
```

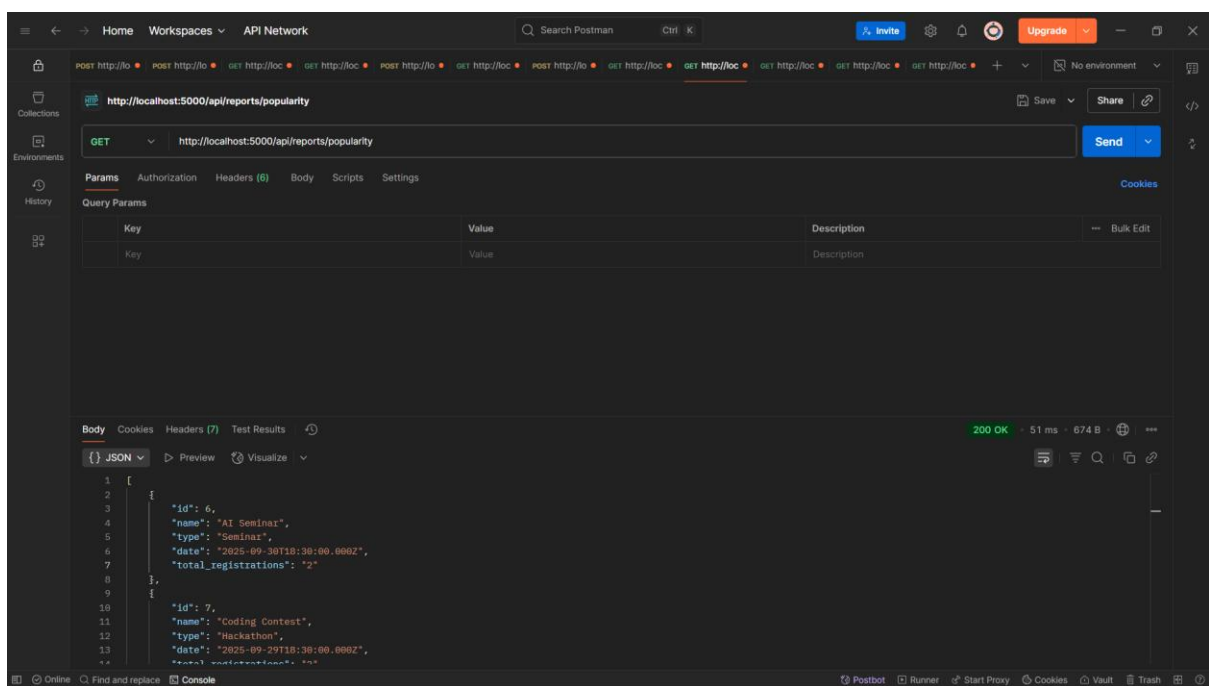


```
[  
  {  
    "id": 1,  
    "student_id": 1,  
    "event_id": 1,  
    "rating": 4  
  },  
  {  
    "id": 3,  
    "student_id": 11,  
    "event_id": 6,  
    "rating": 4  
  },  
  {  
    "id": 4,  
    "student_id": 13,  
    "event_id": 7,  
    "rating": 4  
  },  
  {  
    "id": 5,  
    "student_id": 14,  
    "event_id": 7,  
    "rating": 5  
  }  
]
```

```

    "id": 5,
    "student_id": 14,
    "event_id": 7,
    "rating": 5
  },
  {
    "id": 2,
    "student_id": 2,
    "event_id": 1,
    "rating": 5
  }
]

```

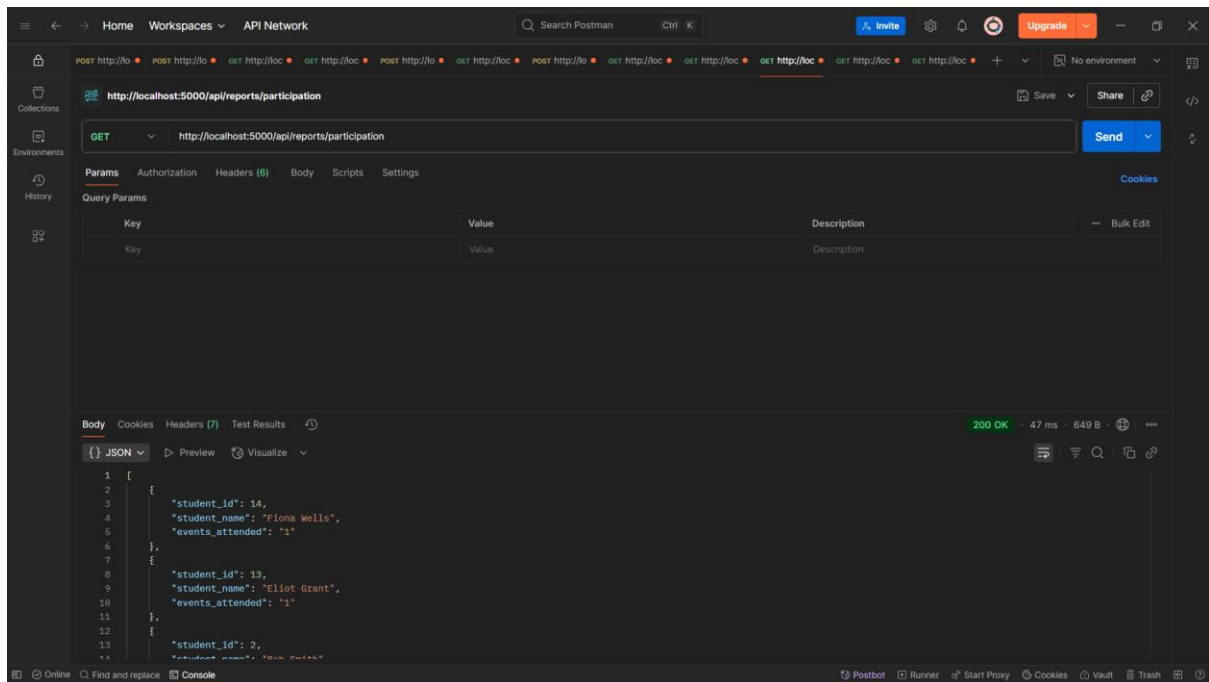


```

[
  {
    "id": 6,
    "name": "AI Seminar",
    "type": "Seminar",
    "date": "2025-09-30T18:30:00.000Z",
    "total_registrations": "2"
  }
]

```

```
},
{
  "id": 7,
  "name": "Coding Contest",
  "type": "Hackathon",
  "date": "2025-09-29T18:30:00.000Z",
  "total_registrations": "2"
},
{
  "id": 1,
  "name": "Hackathon 2025",
  "type": "Workshop",
  "date": "2025-09-14T18:30:00.000Z",
  "total_registrations": "2"
},
{
  "id": 5,
  "name": "Tech Fest 2025",
  "type": "Fest",
  "date": "2025-09-19T18:30:00.000Z",
  "total_registrations": "1"
}
]
```

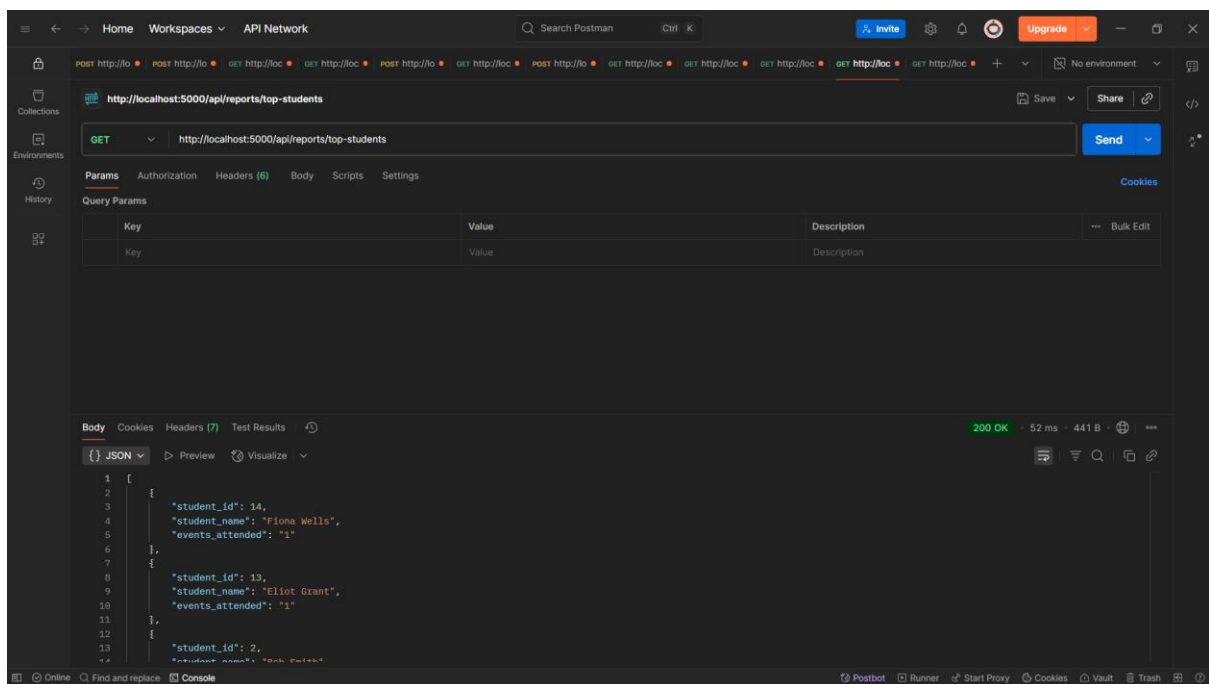


```
[
  {
    "student_id": 14,
    "student_name": "Fiona Wells",
    "events_attended": "1"
  },
  {
    "student_id": 13,
    "student_name": "Eliot Grant",
    "events_attended": "1"
  },
  {
    "student_id": 2,
    "student_name": "Bob Smith",
    "events_attended": "1"
  },
  {
    "student_id": 1,
    "student_name": "Alice Johnson",
    "events_attended": "1"
  }
]
```

```

    },
    {
      "student_id": 11,
      "student_name": "Charlie Davis",
      "events_attended": "1"
    },
    {
      "student_id": 12,
      "student_name": "Dana Lee",
      "events_attended": "0"
    }
  ]
}

```



```

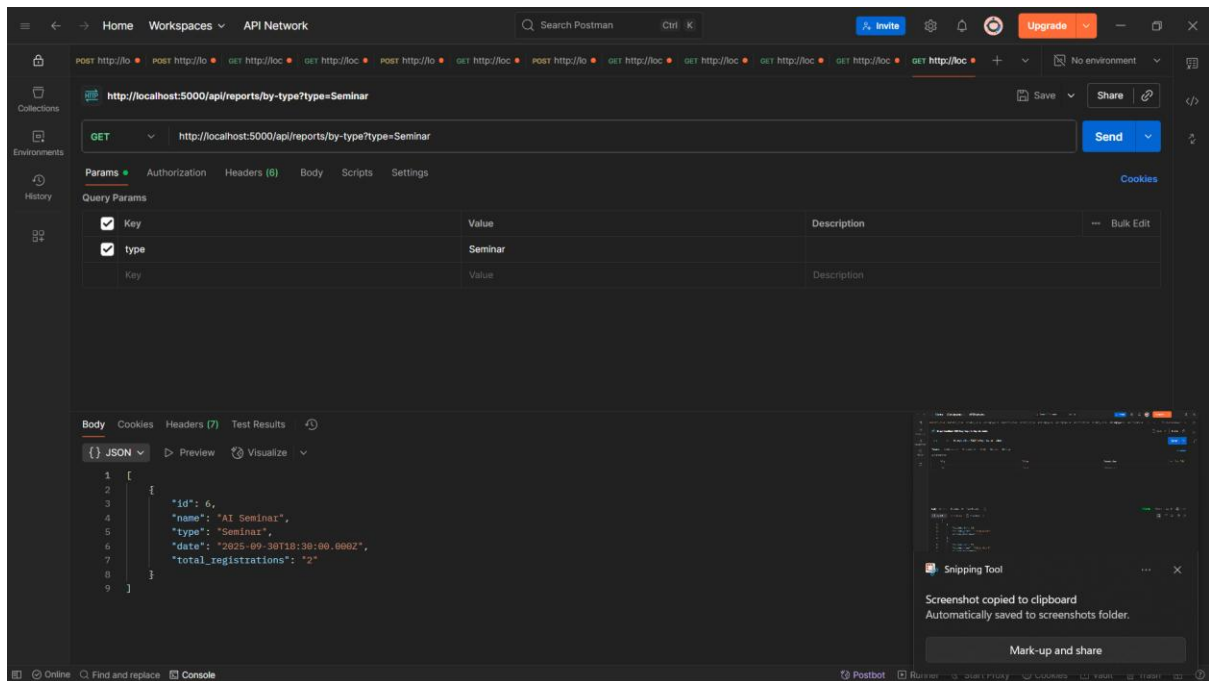
[
  {
    "student_id": 14,
    "student_name": "Fiona Wells",
    "events_attended": "1"
  },
  {

```

```

    "student_id": 13,
    "student_name": "Eliot Grant",
    "events_attended": "1"
  },
  {
    "student_id": 2,
    "student_name": "Bob Smith",
    "events_attended": "1"
  }
]

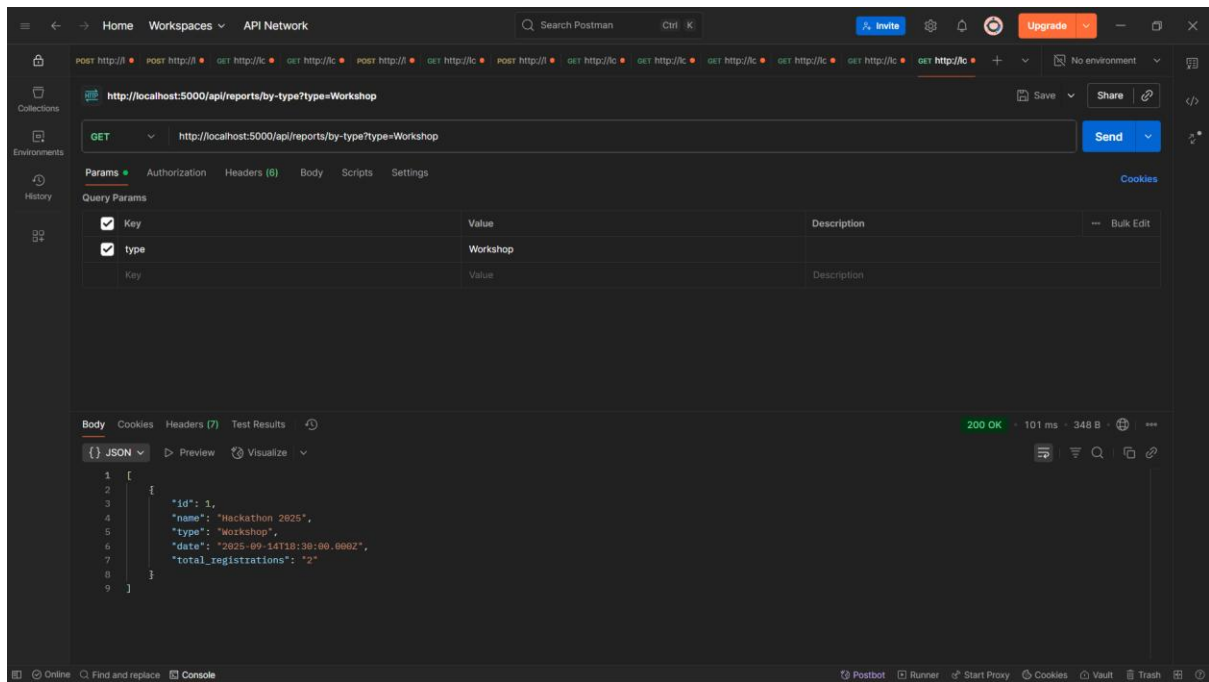
```



```

[
  {
    "id": 6,
    "name": "AI Seminar",
    "type": "Seminar",
    "date": "2025-09-30T18:30:00.000Z",
    "total_registrations": 2
  }
]

```



```
[
  {
    "id": 1,
    "name": "Hackathon 2025",
    "type": "Workshop",
    "date": "2025-09-14T18:30:00.000Z",
    "total_registrations": "2"
  }
]
```