1. **POST /login - User Authentication**
   Description: Logs a user into the system. Supports response in HTML or JSON.
   HTTP Method: POST
   Endpoint: /controller.do?action=login
   Optional Headers:

- Accept: application/json or query parameter ?format=json for JSON response.

Request Body (form-urlencoded):

```
email=user@email.com
password=123456
```

Request Body (JSON):

```
{
  "email": "user@email.com",
  "password": "123456"
}
```

Response Codes:

- **200 OK**
  Successful login (HTML redirected or JSON returned with user data).

Response JSON:

```
{
  "message": "Login successful",
  "userName": "John Doe",
  "userEmail": "john@email.com"
}
```

- **302 Found**
  Redirects to the main page (transactions-page) when login is via form submission.

- **401 Unauthorized**
  Invalid login (incorrect credentials).

Error JSON:

```
{
  "error": "Invalid login"
}
```

- **500 Internal Server Error**
  Internal error while trying to authenticate the user.

---

2. **POST /register - Create New User**
   Description: Registers a new user in the system. Supports response in HTML or JSON.
   HTTP Method: POST
   Endpoint: /controller.do?action=register
   Optional Headers:

- Accept: application/json or query parameter ?format=json for JSON response.

Request Body (form-urlencoded):

```
name=John Doe
email=john@email.com
password=123456
```

Request Body (JSON):

```
{
  "name": "John Doe",
  "email": "john@email.com",
  "password": "123456"
}
```

Response Codes:

- **201 Created** – User successfully registered.
  Success JSON:

```
{
  "message": "User registered successfully.",
  "userId": 42
}
```

- **302 Found** – Redirect to transactions page upon success via HTML form.

- **409 Conflict** – Email already in use.
  Error JSON:

```
{
  "error": "Email already in use."
}
```

- **500 Internal Server Error** – Error trying to register the user.
  Error JSON:

```
{
  "error": "Failed to complete the registration. Please try again."
}
```

---

3. **GET /logout - End User Session**
   Description: Ends the authenticated user's session. Supports response in HTML or JSON.
   HTTP Method: GET
   Endpoint: /controller.do?action=logout
   Optional Headers:

- Accept: application/json or ?format=json for JSON response.

Response Codes:

- **200 OK** – Logout successfully completed.
  Success JSON:

```
{
  "message": "Logout successful"
}
```

- **200 OK (HTML)** – Redirects to index.jsp after logout.

---

4. **GET /transactions - List User Transactions**
   Description: Returns a list of transactions made by the logged-in user, with support for filters and pagination. Response in JSON or HTML.
   HTTP Method: GET
   Endpoint: /controller.do?action=transactions-page
   Query Parameters:

- year (optional): Year to filter transactions (e.g., 2025)
- month (optional): Month to filter transactions (1 to 12)
- type (optional): Transaction type (INCOME or EXPENSE)
- category (optional): Transaction category (e.g., FOOD, ENTERTAINMENT, HEALTH, etc.)
- page (optional): Page number (default: 1)
- size (optional): Items per page (default: 10)

Optional Headers:

- Accept: application/json or ?format=json (not implemented in this command) for JSON response.

Example JSON response (200 OK):


```
{
  "transactions": [
    {
      "id": 42,
      "payerId": 1,
      "receiverId": 3,
      "price": 100.0,
      "description": "Market purchase",
      "type": "EXPENSE",
      "category": "FOOD",
      "dateTime": "2025-07-13T10:45:00"
    }
  ],
  "currentPage": 1,
  "totalPages": 5
}
```

Response Codes:

- **200 OK** – Transactions list returned successfully.
- **400 Bad Request** – Invalid parameter (e.g., year, month, type, category).
- **401 Unauthorized** – User not authenticated.
- **500 Internal Server Error** – Internal error fetching data.

---

5. **GET /transactions-summary - User Financial Summary**
   Description: Returns a summary of the logged-in user's financial transactions, including total income, expenses, current balance, and grouping by category.
   HTTP Method: GET
   Endpoint: /controller.do?action=transactions-summary-page
   Optional Headers:

- Accept: application/json — to get JSON response.

Example JSON response (200 OK):

```
{
  "totalIncome": 5000.00,
  "totalExpenses": 3200.00,
  "currentBalance": 1800.00,
  "expensesByCategory": {
    "FOOD": 1200.00,
    "RENT": 1500.00,
    "ENTERTAINMENT": 500.00
  },
  "incomeByCategory": {
    "SALARY": 4000.00,
    "FREELANCE": 1000.00
  }
}
```

Response Codes:

- **200 OK** – Summary returned successfully.
- **401 Unauthorized** – User not authenticated.

- **500 Internal Server Error** – Error fetching summary from database.

---

6. **POST /transactions - Create New Transaction**
   Description: Registers a new financial transaction between users. User must be authenticated.
   HTTP Method: POST
   Endpoint: /controller.do?action=new-transaction
   Optional Headers:

- Accept: application/json — to receive response in JSON.

Request Body (JSON):

```
{
  "receiverEmail": "user2@example.com",
  "price": "250.75",
  "description": "Book purchase",
  "type": "EXPENSE",
  "category": "EDUCATION"
}
```

It is also possible to send via application/x-www-form-urlencoded form with the same fields.

Response Codes:

- **201 Created** – Transaction created successfully.
- **400 Bad Request** – Missing fields, invalid values, or attempt to transfer to own email.
- **401 Unauthorized** – User not authenticated.
- **404 Not Found** – Recipient email not found.
- **500 Internal Server Error** – Error saving the transaction.

---

7. **GET /transactions/edit - Retrieve Transaction Data for Editing**
   Description: Retrieves data of a specific transaction for editing. Returns data in JSON format or shows the edit screen depending on Accept header.
   HTTP Method: GET
   Endpoint: /controller.do?action=edit-transaction-page

Query Parameters:

- id (required) — ID of the transaction to retrieve.

Optional Headers:

- Accept: application/json — for JSON response.

Example request:

```
GET /controller.do?action=edit-transaction-page&id=5
Accept: application/json
```

Response (JSON):

```
{
  "id": 5,
  "payerId": 1,
  "receiverId": 2,
  "price": 250.75,
  "description": "Book purchase",
  "type": "EXPENSE",
  "category": "EDUCATION",
  "dateTime": "2025-07-13T10:23:00",
  "receiverEmail": "user2@example.com"
}
```

Response Codes:

- **200 OK** – Transaction found successfully.
- **400 Bad Request** – Missing or invalid id parameter.
- **404 Not Found** – Transaction not found.
- **500 Internal Server Error** – Unexpected server error.

---

8. **PUT /transactions - Edit Transaction**
   Description: Updates data of an existing transaction. Available in JSON format and also via HTML form.

HTTP Method: PUT
Endpoint: /controller.do?action=edit-transaction
Request Body (JSON):

```
{
  "id": 5,
  "receiverEmail": "user2@example.com",
  "price": "150.50",
  "description": "Monthly payment",
  "type": "EXPENSE",
  "category": "HOUSING"
}
```

All fields are mandatory.

Success response (JSON):

```
{
  "message": "Transaction updated successfully."
}
```

Response Codes:

- **200 OK** – Transaction updated successfully.
- **400 Bad Request** – Missing fields, invalid data, or unauthorized operation.
- **401 Unauthorized** – User not authenticated.
- **404 Not Found** – Transaction or recipient not found.
- **500 Internal Server Error** – Unexpected server error.

---

9. **DELETE /transactions - Delete Transaction**
   Description: Removes a transaction registered by the authenticated user.
   HTTP Method: DELETE
   Endpoint: /controller.do?action=delete-transaction
   Request Body (JSON):

```
{
```

```
  "id": 5
}
```

Alternatively, the ID can also be sent as a URL parameter:

```
/controller.do?action=delete-transaction&id=5
```

Response Codes:

- **204 No Content** – Transaction deleted successfully.
- **400 Bad Request** – Invalid or missing ID.
- **401 Unauthorized** – User not authenticated.
- **403 Forbidden** – Attempt to delete a transaction belonging to another user.
- **404 Not Found** – Transaction not found.
- **500 Internal Server Error** – Failure deleting the transaction.

10. GET /transactions/id - Get Transaction by ID

Description: Returns the data of a specific transaction belonging to the authenticated user.
HTTP Method: GET
Endpoint: `/transacoes/id?id={id}`

Query Parameters:

- `id` (long): ID of the transaction to retrieve.

Expected Headers:

- `Accept: application/json`
- `Cookie: JSESSIONID=...` (for authenticated session)

Example Request:

```
GET /transacoes/id?id=123
Accept: application/json
```

Response (JSON):

```
{
  "id": 123,
  "payerId": 1,
  "receiverId": 2,
  "price": 150.00,
  "description": "Rent",
  "type": "EXPENSE",
  "category": "HOUSING",
  "dateTime": "2025-07-13T10:40:00",
  "receiverEmail": "example@domain.com"
}
```


Response Codes:

- **200 OK** – Transaction found successfully.
- **400 Bad Request** – Invalid or missing ID.
- **401 Unauthorized** – User not authenticated.
- **403 Forbidden** – Attempt to access a transaction belonging to another user.
- **`500 Internal Server Error`** – Erro inesperado no servidor.