#### **Software Requirements Specification (SRS)**

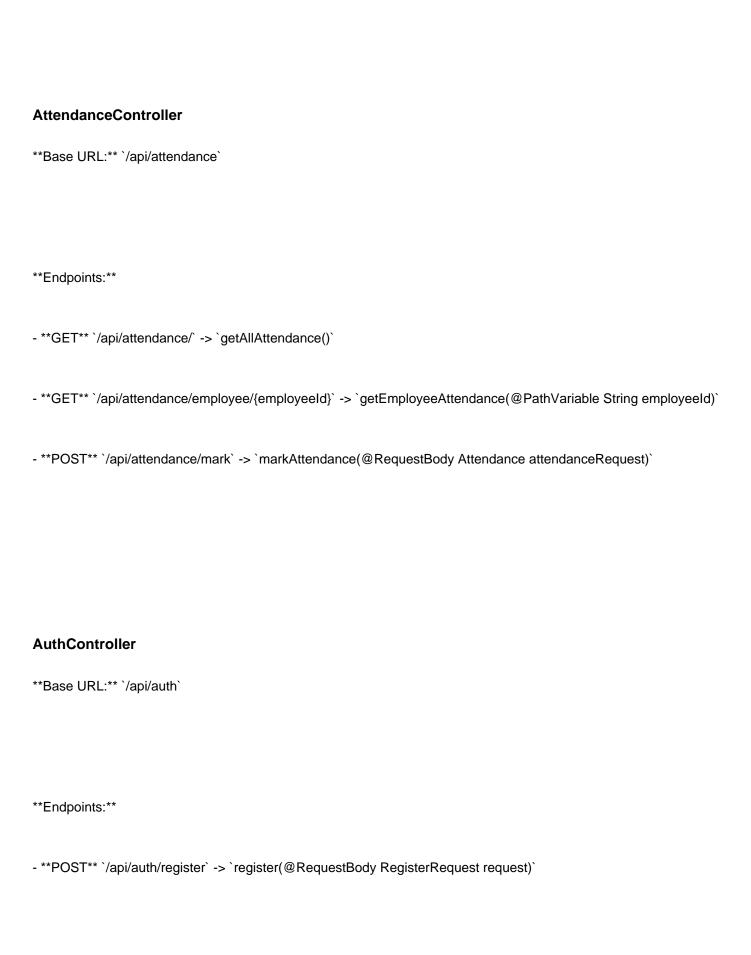
#### Overview

This document describes the API endpoints and responsibilities of the controller classes in the backend application.

#### **AssetController**

```
**Base URL:** \data{api/assets}
```

- \*\*Endpoints:\*\*
- \*\*GET\*\* `/api/assets/` -> `getAllAssets()`
- \*\*GET\*\* `/api/assets/category/{category}` -> `getAssetsByCategory(@PathVariable String category)`
- \*\*GET\*\* `/api/assets/status/{status}` -> `getAssetsByStatus(@PathVariable String status)`
- \*\*GET\*\* `/api/assets/search` -> `searchAssetsByAssignedTo(@RequestParam String keyword)`
- \*\*GET\*\* `/api/assets/employee/{id}` -> `getAssetsByEmployeeDbId(@PathVariable Long id)`
- \*\*POST\*\* `/api/assets/` -> `addAsset(@RequestBody Asset asset)`
- \*\*PUT\*\* `/api/assets/{id}` -> `updateAsset(@PathVariable Long id, @RequestBody Asset asset)`
- \*\*DELETE\*\* `/api/assets/{id}` -> `deleteAsset(@PathVariable Long id)`

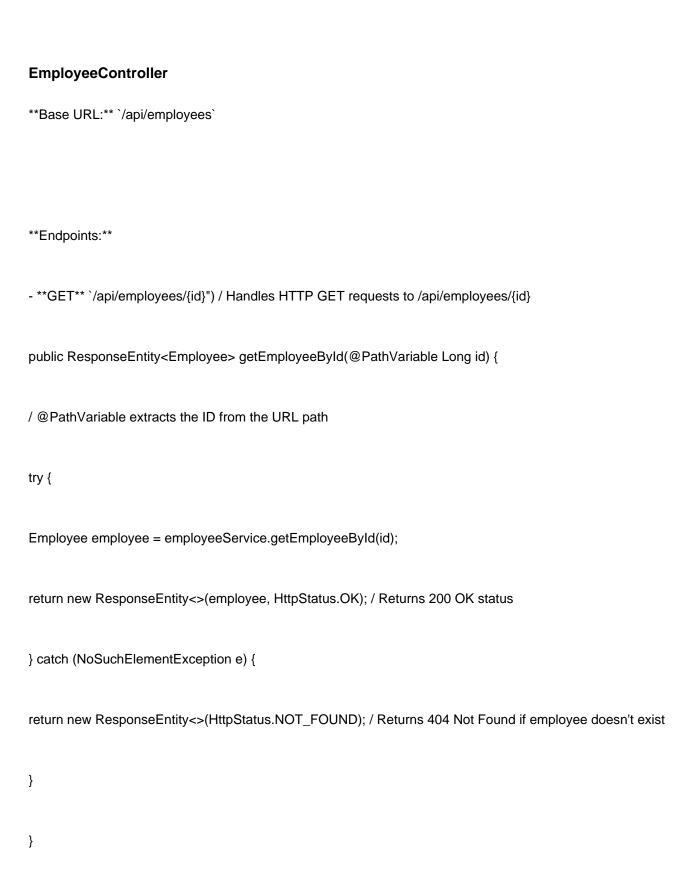








- **GET** `/api/companyregistrations/` -> `getAllCompanyRegistrations()`
- **GET** `/api/companyregistrations/{id}` -> `getCompanyRegistrationById(@PathVariable Long id)`
- **POST** `/api/companyregistrations/` -> `createCompanyRegistration(@RequestBody CompanyRegistration companyRegistration)`
- **PUT** `/api/companyregistrations/{id}` -> `updateCompanyRegistration(@PathVariable Long id, @RequestBody CompanyRegistration companyRegistrationDetails)`
- **DELETE** `/api/companyregistrations/{id}` -> `deleteCompanyRegistration(@PathVariable Long id)`
ElectricBillsController
**Base URL:** `/api/electric-bills`
**Endpoints:**
- **POST** `/api/electric-bills/` -> `createElectricBill(@RequestBody ElectricBillsDTO electricBillsDTO)`
- **GET** `/api/electric-bills/` -> `getAllElectricBills()`
- **PUT** `/api/electric-bills/{id}` -> `updateElectricBill(@PathVariable Long id, @RequestBody ElectricBillsDTO electricBillsDTO)`
- **DELETE** `/api/electric-bills/{id}` -> `deleteElectricBill(@PathVariable Long id)`





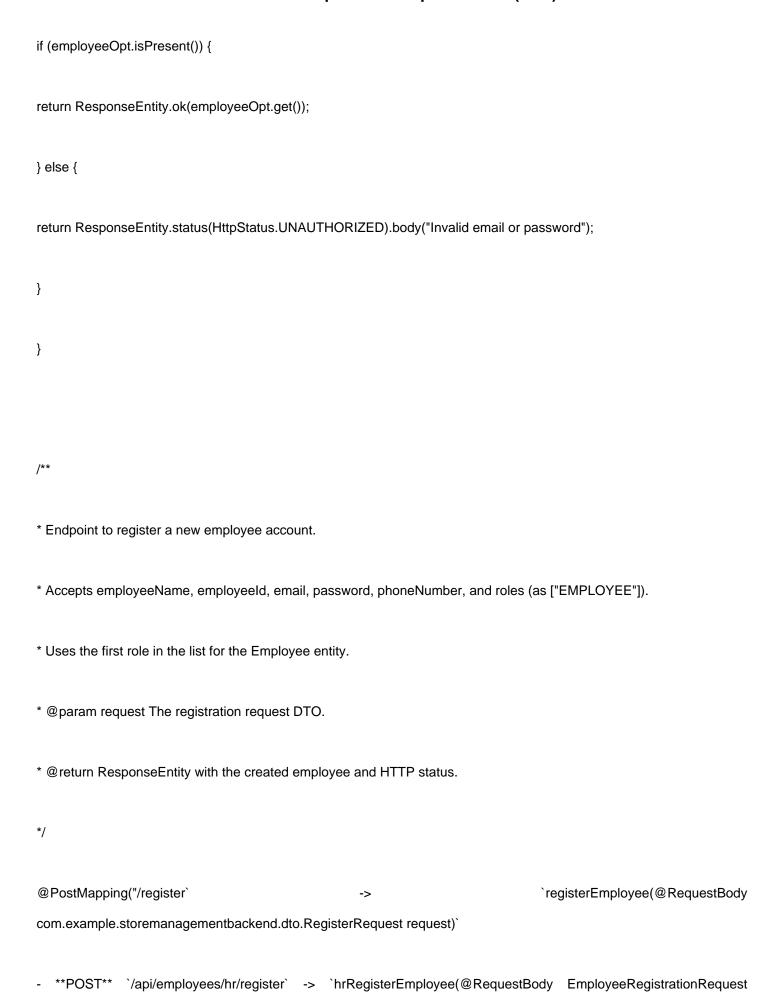
/**
* Endpoint to get all employee records.
* @return ResponseEntity with a list of all employees and HTTP status.
*/
@GetMapping / Handles HTTP GET requests to /api/employees
public ResponseEntity <list<employee>&gt; getAllEmployees() {</list<employee>
List <employee> employees = employeeService.getAllEmployees();</employee>
return new ResponseEntity<>(employees, HttpStatus.OK);
}
/* <i>*</i>
* Endpoint to update an existing employee record.
* @param id The internal ID of the employee to update.
* @param employeeDetails The updated employee details.
* @return ResponseEntity with the updated employee and HTTP status.

@PutM	lapping("/{id}") / Handles HTTP I	PUT requests to /api/employ	ees/{id}				
public employ	ResponseEntity <employee></employee>	updateEmployee(@PathV	ariable	Long	id,	@RequestBody	Employee
try {							
Employ	vee updatedEmployee = employo	eeService.updateEmployee(	id, emplo	oyeeDeta	ails);		
return ı	new ResponseEntity<>(updatedl	Employee, HttpStatus.OK); /	Returns	200 OK			
} catch	(NoSuchElementException e) {						
return ı	new ResponseEntity<>(HttpState	us.NOT_FOUND); / Returns	404 Not	Found			
}							
}							
/**							
* Endp	oint to delete an employee recor	d.					
* @par	am id The internal ID of the emp	oloyee to delete.					
* @reti	urn ResponseEntity with HTTP s	tatus.					
*/							

 $@\,DeleteMapping("/\{id\}")\,/\,Handles\,HTTP\,\,DELETE\,\,requests\,\,to\,/api/employees/\{id\}$ 

public ResponseEntity <httpstatus> deleteEmployee(@PathVariable Long id) {</httpstatus>
try {
employeeService.deleteEmployee(id);
return new ResponseEntity<>(HttpStatus.NO_CONTENT); / Returns 204 No Content (successful deletion)
} catch (NoSuchElementException e) {
return new ResponseEntity<>(HttpStatus.NOT_FOUND); / Returns 404 Not Found
}
}
/* <i>*</i>
* Endpoint to update an employee's status.
* @param id The internal ID of the employee to update.
* @param status The new status for the employee.
* @return ResponseEntity with the updated employee and HTTP status.
*/
@PutMapping("/{id}/status") / Handles HTTP PUT requests to /api/employees/{id}/status

public ResponseEntity <employee> updateEmployeeStatus(@PathVariable Long id, @RequestParam String status)</employee>
try {
Employee updatedEmployee = employeeService.updateEmployeeStatus(id, status);
return new ResponseEntity<>(updatedEmployee, HttpStatus.OK);
} catch (NoSuchElementException e) {
return new ResponseEntity<>(HttpStatus.NOT_FOUND);
}
}
@PostMapping("/login")
<pre>public ResponseEntity<?> login(@RequestBody Map<string, string=""> loginData) {</string,></pre>
String email = loginData.get("email");
String password = loginData.get("password");
Optional <employee> employeeOpt = employeeService.getAllEmployees().stream()</employee>
.filter(emp -> emp.getEmail().equals(email) && emp.getPassword().equals(password))
.findFirst();



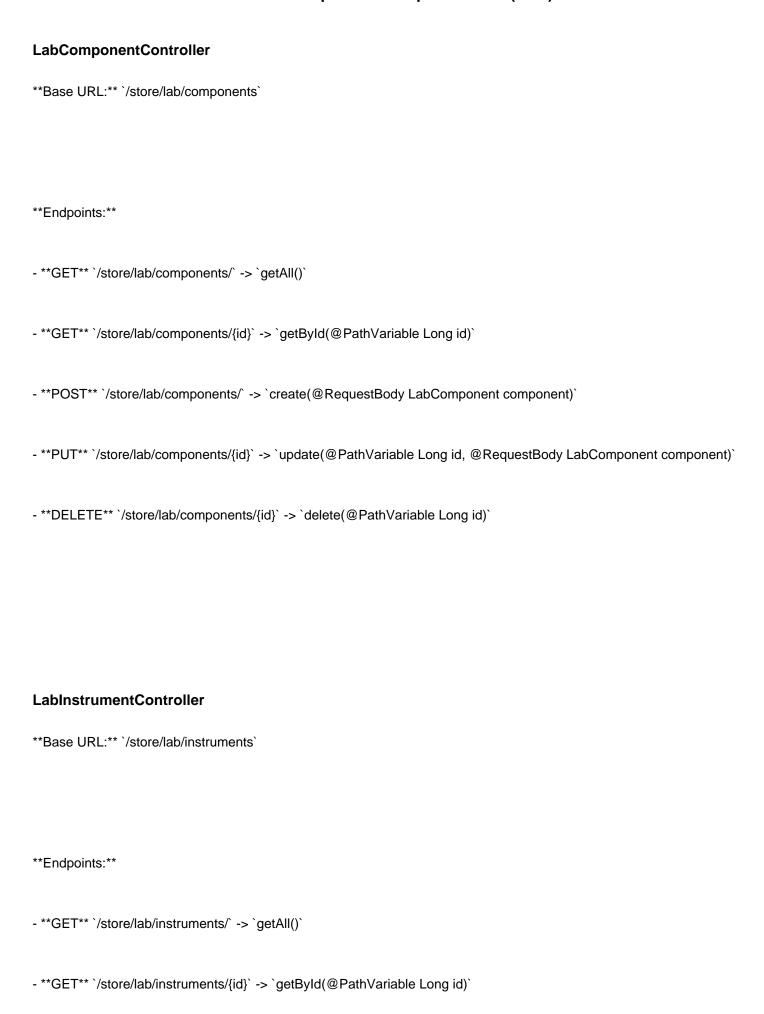


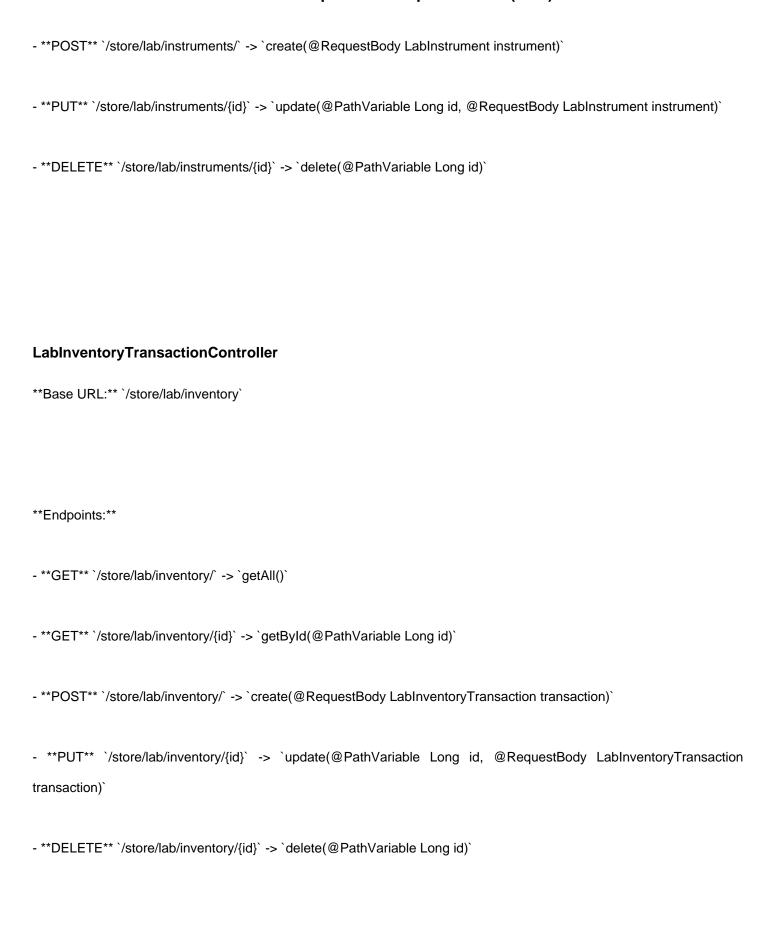


- **GET** /api/financereports/{id}* -> *getFinanceReportById(@PathVariable Long id)
- **POST** `/api/financereports/` -> `createFinanceReport(@RequestBody FinanceReport financeReport)`
- **PUT** `/api/financereports/{id}` -> `updateFinanceReport(@PathVariable Long id, @RequestBody FinanceReport financeReportDetails)`
- **DELETE** `/api/financereports/{id}` -> `deleteFinanceReport(@PathVariable Long id)`
FixedStationaryController
**Base URL:** `/store/stationary/fixed`
**Endpoints:**
- **GET** `/store/stationary/fixed/` -> `getAll()`
- **GET** `/store/stationary/fixed/{id}` -> `getById(@PathVariable Long id)`
- **POST** `/store/stationary/fixed/` -> `create(@RequestBody FixedStationary stationary)`
- **PUT**`/store/stationary/fixed/{id}`-> `update(@PathVariable Long id, @RequestBody FixedStationary stationary)`
- **DELETE** `/store/stationary/fixed/{id}` -> `delete(@PathVariable Long id)`



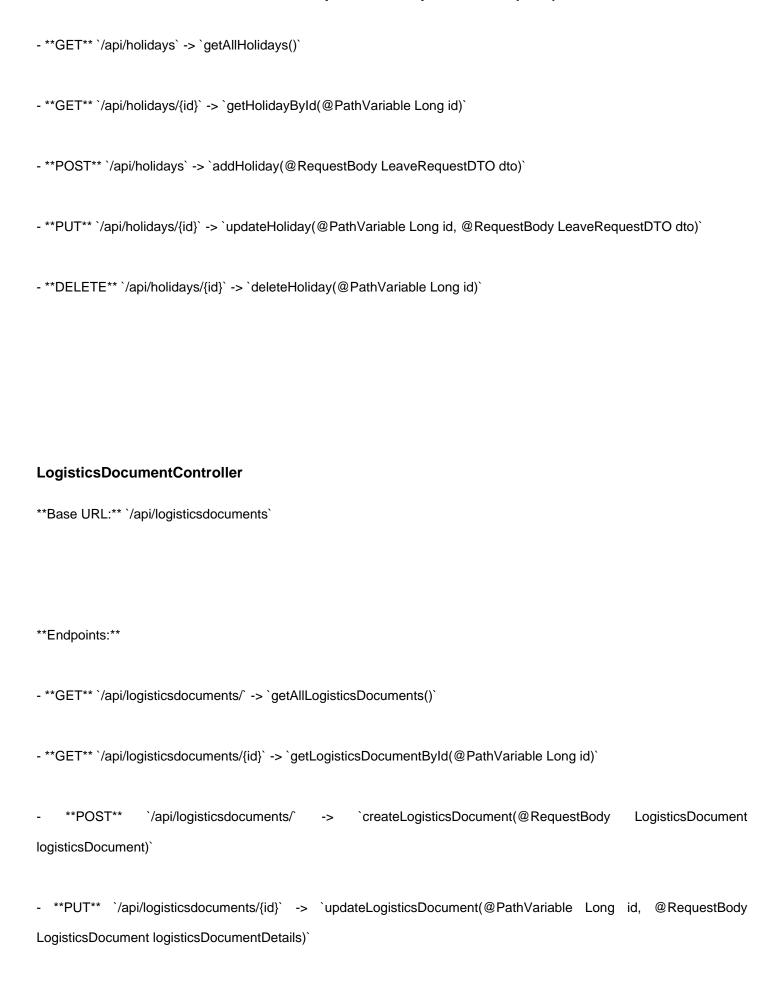
- **GET** `/a	pi/incentives/`->`ge	tAllIncentives	s()`					
- **PUT** incentivesDT	`/api/incentives/{id}` -O)`	-> `updat	teIncentive(@Pa	nthVariable	Long	id, (	@RequestBody	IncentivesDTC
- **DELETE*	* `/api/incentives/{id}	`->`deleteIn	centive(@PathV	′ariable Long	g id)`			
	IsController  ** `/api/internet-bills`							
**Endpoints:	**							
- **POST** `/	/api/internet-bills/`->	`createIntern	netBill(@Request	tBody Intern	etBillsD	TO int	ternetBillsDTO)`	
- **GET** `/a	pi/internet-bills/` -> `ç	getAllInternet	Bills()`					
- **PUT** ` internetBillsD	·/api/internet-bills/{id}	`-> `update	eInternetBill(@Pa	'athVariable	Long	id, @	RequestBody	InternetBillsDTC
- **DELETE*	** `/api/internet-bills/{i	d}` -> `delete	eInternetBill(@Pa	athVariable L	Long id)`			





# LabMaterialController \*\*Base URL:\*\* `/store/lab/materials` \*\*Endpoints:\*\* - \*\*GET\*\* `/store/lab/materials/` -> `getAll()` - \*\*GET\*\* `/store/lab/materials/{id}` -> `getById(@PathVariable Long id)` - \*\*POST\*\* `/store/lab/materials/` -> `create(@RequestBody LabMaterial material)` - \*\*PUT\*\* `/store/lab/materials/{id}` -> `update(@PathVariable Long id, @RequestBody LabMaterial material)` - \*\*DELETE\*\* `/store/lab/materials/{id}` -> `delete(@PathVariable Long id)` LeaveController \*\*Base URL:\*\* `/api/leaves` \*\*Endpoints:\*\* - \*\*GET\*\* `/api/leaves/` -> `getAllLeaves()` - \*\*POST\*\* `/api/leaves/` -> `createLeave(@RequestBody Leave leave)`







\*\*Base URL:\*\* `/api/performance-reviews`

**Endpoints:**
- **GET** `/api/performance-reviews/{id}") / Handles HTTP GET requests to /api/performance-reviews/{id}
public ResponseEntity <performancereview> getPerformanceReviewById(@PathVariable Long id) {</performancereview>
try {
PerformanceReview review = performanceReviewService.getPerformanceReviewByld(id);
return new ResponseEntity<>(review, HttpStatus.OK); / Returns 200 OK status
} catch (NoSuchElementException e) {
return new ResponseEntity<>(HttpStatus.NOT_FOUND); / Returns 404 Not Found if review doesn't exist
}
}
/**
* Endpoint to get all performance reviews in the system.
* @return ResponseEntity with a list of all performance reviews and HTTP status.
*/

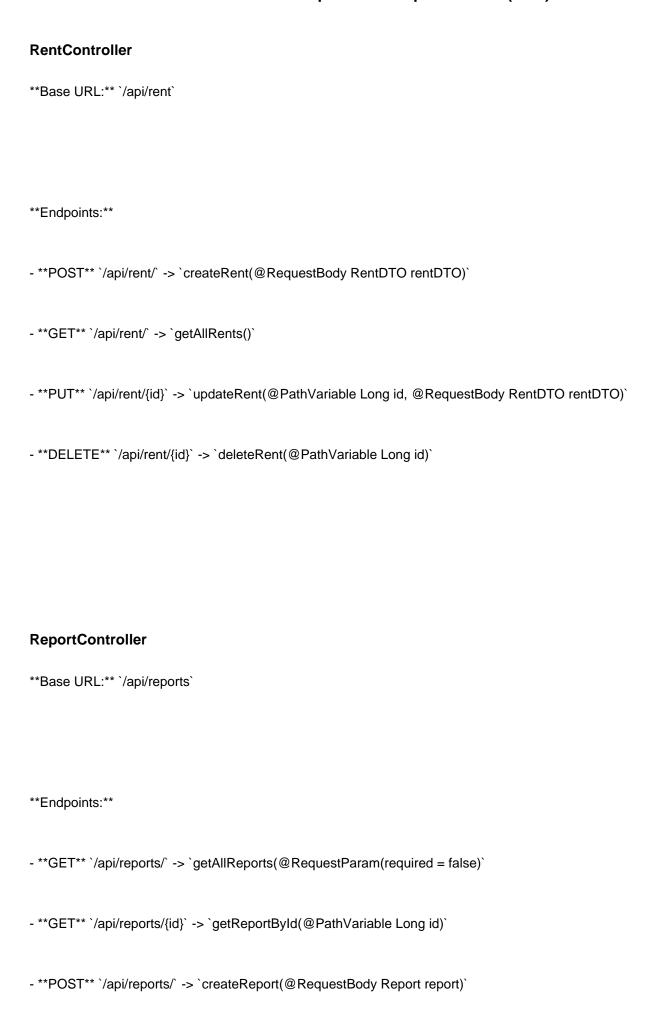
@GetMapping / Handles HTTP GET requests to /api/performance-reviews	
public ResponseEntity <list<performancereview>&gt; getAllPerformanceReviews() {</list<performancereview>	
List <performancereview> reviews = performanceReviewService.getAllPerformanceReviews();</performancereview>	
return new ResponseEntity<>(reviews, HttpStatus.OK);	
}	
/**	
* Endpoint to get all performance reviews for a specific employee.	
* This is useful for displaying all reviews on an employee's profile page.	
* @param employeeId The ID of the employee.	
* @return ResponseEntity with a list of performance reviews for the employee and HTTP status.	
*/	
@GetMapping("/employee/{employeeId}") / Handles HTTP GET requests /api/performance-reviews/employee/{employeeId}	tc
public ResponseEntity <list<performancereview>&gt; getPerformanceReviewsByEmployeeId(@PathVariable S employeeId) {</list<performancereview>	String
try {	

List <performancereview> reviews = performanceReviewService.getPerformanceReviewsByEmployeeId(employeeId)</performancereview>
return new ResponseEntity<>(reviews, HttpStatus.OK);
} catch (NoSuchElementException e) {
return new ResponseEntity<>(HttpStatus.NOT_FOUND); / Employee not found or no reviews for employee
}
}
/**
* Endpoint to get all performance reviews for a specific employee by numeric ID.
* @param id The internal numeric ID of the employee.
* @return ResponseEntity with a list of performance reviews for the employee and HTTP status.
*/
@GetMapping("/employee/byId/{id}` -> `getPerformanceReviewsByEmployeeDbId(@PathVariable Long id)`

### PrinterController

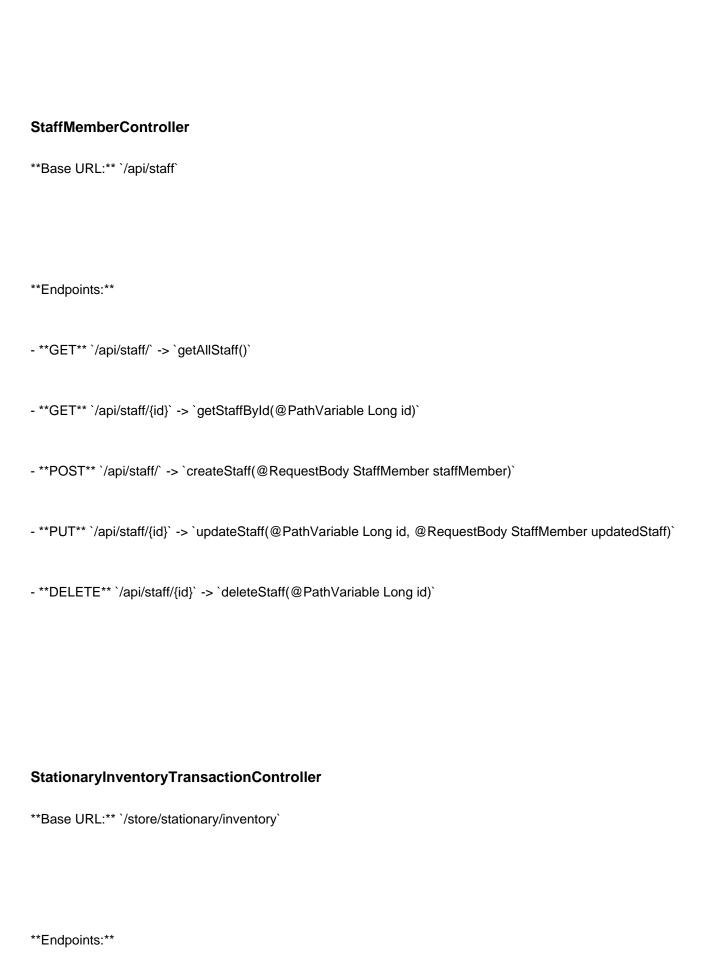


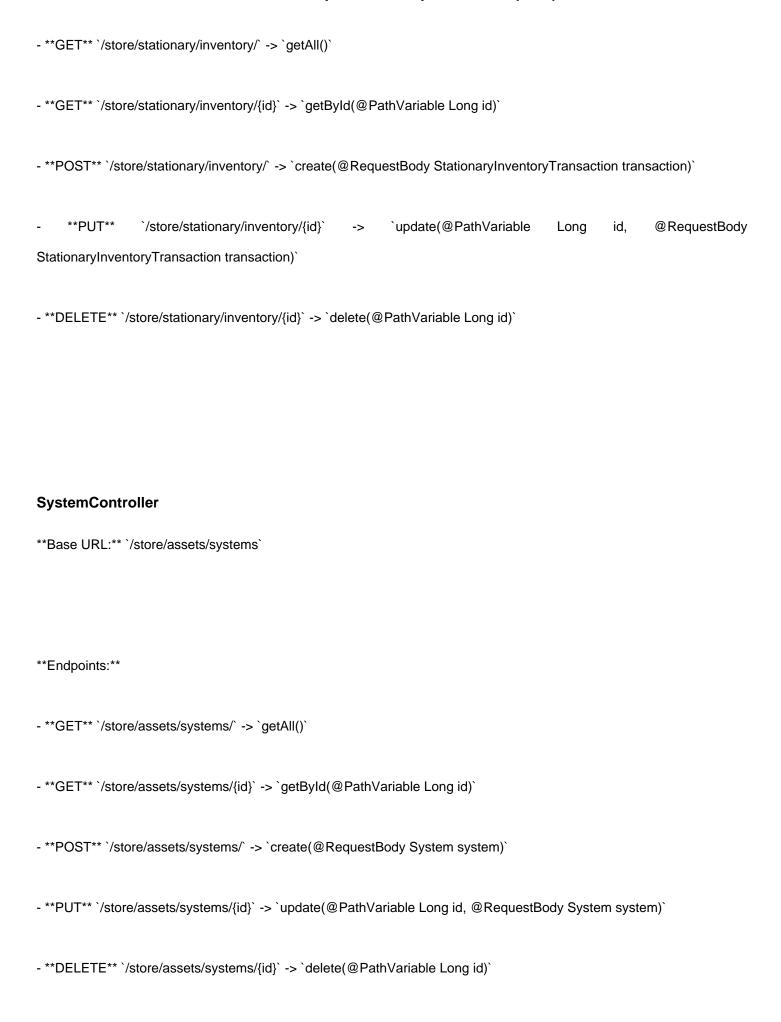


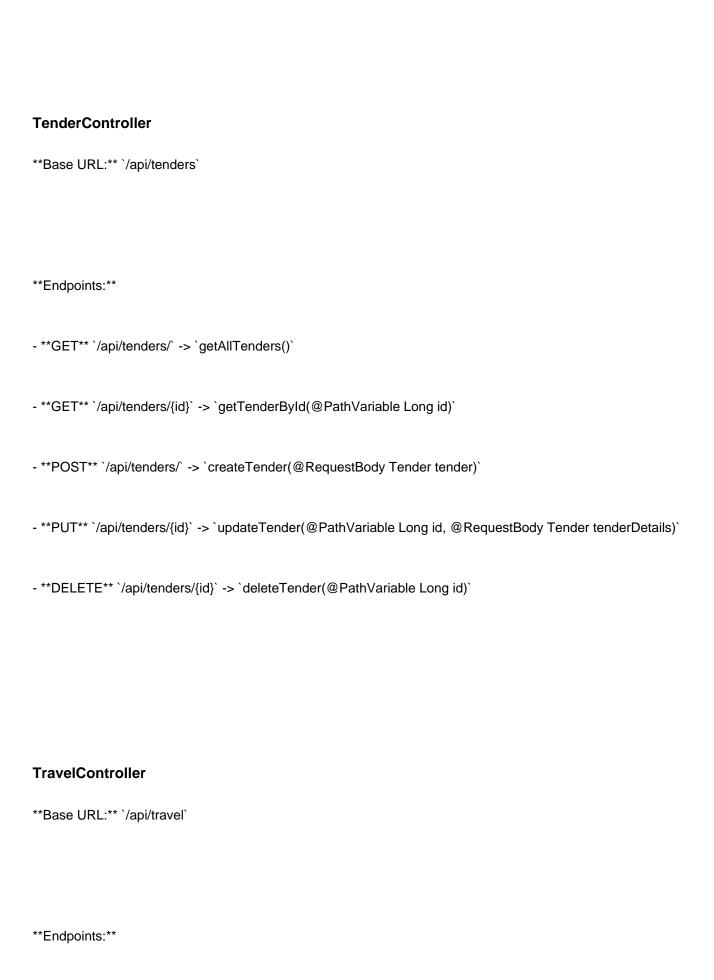














WeeklyActivityController
**Base URL:** `/api/activities`
**Endpoints:**