# BQS API Migration Documentation (Story IHS3-468)

**📘 BQS API Migration Documentation (Story IHS3-468)**

**✅ Overview**

Migration from old department-specific API endpoints to a unified, header-driven generic API for better modularity, scalability, and maintainability.

**🔁 Old vs New API Structure**

| Aspect | Old API (`/getPrefillData`) | New API (`/bqs-prefill-data`) |

|--------|-----------------------------|---------------------------------|

| Routing Logic | Hardcoded in Controller | Header-based dynamic routing |

| Endpoint Style | One for all types | Unified generic endpoint |

| Headers Used | None | `X-SYF-API-ID`, `X-SYF-API-KEY` |

| Response Format | Direct object | `GenericResponse<T>` wrapper |

| Error Handling | Manual | Global Exception Handler |

| Proxy Structure | Single hardwired proxy | Modular multiple proxies |

| Scalability | Poor | Easy to onboard new departments |

📌 Old Endpoint

Endpoint: `/getPrefillData`

Method: POST

Sample Controller:

@PostMapping("/getPrefillData")

public ResponseEntity<PrefillResponse> getData(@RequestBody PrefillRequest request) {

return ResponseEntity.ok(service.fetchData(request));

}

Drawbacks:

- Tight coupling

- if-else based routing

- Difficult to maintain & scale

**🚀 New Endpoint Architecture**

**✅ Unified Endpoint**

Endpoint: `/bqs-prefill-data`

Method: POST

Controller Code:

@PostMapping("/bqs-prefill-data")

public ResponseEntity<GenericResponse<PrefillResponse>> getPrefillData(

@RequestBody @Valid PrefillRequest request,

@RequestHeader("X-SYF-API-ID") String apiId) {

switch (apiId) {

case "AUTO": return autoProxy.getPrefillData(request);

case "DENTAL": return dentalProxy.getPrefillData(request);

case "HOSPITAL": return hospitalProxy.getPrefillData(request);

default: throw new IllegalArgumentException("Unsupported API ID");

}

}

**🧠 Key Features**

- Header-based routing using `X-SYF-API-ID`

- Common endpoint with modular proxy classes

- Uses `GenericResponse<T>` for consistent API responses

**📦 Generic Response Wrapper**

public class GenericResponse<T> {

private String status; // SUCCESS / FAILURE

private T data;

private String errorCode;

private String message;

}

**📥 Request Examples by Department**

**🟢 GENERAL**

Header: X-SYF-API-ID: GENERAL

{

"applicationId": "12345",

"state": "CA",

"ssn": "123-45-6789"

}

Response:

{

"status": "SUCCESS",

"data": {

"patientId": "P12345",

"fullName": "John Doe",

"dob": "1990-01-01"

}

}

**🟢 DENTAL**

Header: X-SYF-API-ID: DENTAL

{

"applicationId": "DENTAL\_001",

"toothCode": "T12"

}

Response:

{

"status": "SUCCESS",

"data": {

"patientName": "Jane Smith",

"toothCode": "T12",

"treatment": "Cavity Fill"

}

}

**🟢 AUTO**

Header: X-SYF-API-ID: AUTO

{

"applicationId": "AUTO\_001",

"licensePlate": "MH12XY1234"

}

Response:

{

"status": "SUCCESS",

"data": {

"owner": "Rahul Kumar",

"vehicle": "Honda City",

"year": 2020

}

}

**📑 DTO with Validations**

PrefillRequest.java

public class PrefillRequest {

@JsonProperty("applicationId")

@NotBlank(message = "Application ID is mandatory")

private String applicationId;

@JsonProperty("state")

@NotBlank(message = "State is required")

private String state;

@JsonProperty("ssn")

@Pattern(regexp = "\d{3}-\d{2}-\d{4}", message = "SSN must follow XXX-XX-XXXX format")

private String ssn;

@JsonProperty("dob")

@Pattern(regexp = "\d{4}-\d{2}-\d{2}", message = "DOB must be in YYYY-MM-DD format")

private String dob;

private String firstName;

private String lastName;

}

PrefillResponse.java

public class PrefillResponse {

private String patientId;

private String fullName;

private String dob;

private String department;

}

**🔐 Required Headers**

- X-SYF-API-ID: Department type (e.g., AUTO, DENTAL)

- X-SYF-API-KEY: (Optional) API key for security

- Content-Type: application/json

**📘 Swagger Annotations**

@Tag(name = "Patient Prefill API", description = "Endpoints for fetching prefill data")

@Operation(summary = "Fetch Prefill Data", description = "Returns department-wise prefill data")

@Parameters({

@Parameter(name = "X-SYF-API-ID", description = "Department ID", required = true),

@Parameter(name = "X-SYF-API-KEY", description = "Auth Key", required = false)

})

@ApiResponses({

@ApiResponse(responseCode = "200", description = "Success"),

@ApiResponse(responseCode = "400", description = "Bad Request"),

@ApiResponse(responseCode = "500", description = "Server Error")

})

**⚠️ Global Exception Handler**

@RestControllerAdvice

public class GlobalExceptionHandler {

@ExceptionHandler(IllegalArgumentException.class)

public ResponseEntity<ErrorResponse> handleInvalidHeader(IllegalArgumentException ex) {

return ResponseEntity.badRequest().body(

new ErrorResponse("400\_BAD\_REQUEST", ex.getMessage())

);

}

}

**✅ Summary of Benefits**

- One generic endpoint instead of many

- Easy routing with X-SYF-API-ID

- Modular proxy implementation

- Full Swagger documentation

- DTO validations ensure correct input

- Global exception handler for error consistency

- Easier onboarding of new departments

\_End of Technical Documentation for IHS3-468 Migration\_