

Business Analysis Documentation

Shughail System (شغيل)

Prepared by:

Turki Alharbi

Shughail is a home service management system that connects customers with professional workers such as electricians, plumbers, cleaners, and other service providers.

The system allows users to create service requests, describe their needs, and get matched with available workers who can perform the required job efficiently.

The main goal of Shughail is to provide an easy and reliable way to request and manage home services through a structured and validated data process.

The system consists of three main models:

1. User – represents the customer who requests the service.
2. Worker – represents the service provider who performs the job.
3. ServiceRequest – represents the service request that links the user with the worker.

Each model includes specific validation rules to ensure that all data entered into the system is accurate, complete, and follows logical constraints.

Model 1: User

1. userId:

- Can not be null.
- Length must be more than 2 characters.

@NotNull(message = "User ID can not be null")

@Size(min = 3, message = "User ID must be at least 3 characters long")

private String userId;

2. name:

- Can not be null.
- Must contain at least 2 characters.

@NotEmpty(message = "Name can not be empty")

@Size(min = 2, message = "Name must contain at least 2 characters")

private String name;

3. email:

- Can not be null.
- Must be a valid email format.

@NotEmpty(message = "Email is required")

@Email(message = "Invalid email format")

private String email;

4. phoneNumber:

- Must start with “05”.
- Must contain exactly 10 digits.

@Pattern(regexp = "05\\d{8}", message = "Phone number must start with 05 and be 10 digits")

private String phoneNumber;

Model 2: Worker

1. workerId:

- Can not be null.

@NotNull(message = "Worker ID can not be null")

private String workerId;

2. name:

- Can not be null.

@NotEmpty(message = "Name is required")

private String name;

3. profession:

- Can not be null.
- Must specify a valid job (e.g., plumber, electrician).

@NotEmpty(message = "Profession must be provided (e.g., plumber, electrician)")

private String profession;

4. experienceYears:

- Can not be negative.
- Can not exceed 50 years.

@Min(value = 0, message = "Experience years can not be negative")

@Max(value = 50, message = "Experience years can not exceed 50")

private int experienceYears;

5. rating:

6.

- Cannot be below 0.0.
- Cannot exceed 5.0.

@DecimalMin(value = "0.0", message = "Rating can not be below 0.0")

@DecimalMax(value = "5.0", message = "Rating can not exceed 5.0")

private double rating;

Model 3: ServiceRequest

1.requestId:

- Can not be null.

@NotNull(message = "Request ID can not be null")

private String requestId;

2. userId:

- Can not be null.

@NotNull(message = "User ID is required")

private String userId;

3. workerId:

- Can not be null.

@NotNull(message = "Worker ID is required")

private String workerId;

4. serviceType:

- Can not be null.

@NotEmpty(message = "Service type must not be empty")

private String serviceType;

5. description:

- Must contain at least 10 characters.

@Size(min = 10, message = "Description must be at least 10 characters long")

private String description;

6. requestDate:

- Can not be in the past.
- Must follow the format “yyyy-MM-dd”.

@FutureOrPresent(message = "Request date can not be in the past")

@JsonFormat(pattern = "yyyy-MM-dd")

private LocalDate requestDate;

7. status:

- Must be one of the following: Pending, Accepted, Completed, or Cancelled.

@Pattern(regexp =
"Pending|Accepted|Completed|Cancelled", message =
"Invalid status value")

private String status;

8. price:

- Cannot be null.
- Must be greater than 0.

@NotNull(message = "Price cannot be null")

@Positive(message = "Price must be greater than 0")

private Double price;