Module Interface Specification for Sayyara

Team 3, Tiny Coders
Arkin Modi
Joy Xiao
Leon So
Timothy Choy

January 16, 2023

1 Revision History

Table 1: Revision History

Date	**Developer(s)**	**Change**
December 28, 2022	Arkin Modi	Create Revision History
January 7, 2023	Joy Xiao	Introduction
January 9, 2023	Arkin Modi	Add Module Hierarchy
January 11, 2023	Arkin Modi	Create MIS of User Module
January 13, 2023	Arkin Modi	Create MIS of Database Driver Module
January 15, 2023	Leon So	Create MIS of Shop Module

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at https://github.com/arkinmodi/project-sayyara/blob/main/docs/SRS/SRS.pdf

[Also add any additional symbols, abbreviations or acronyms —SS]

symbol	description
Sayyara	Explanation of program name
MIS	Module Interface Specifications
MG	Module Guide

Contents

List of Tables

List of Figures

3 Introduction

The following document details the Module Interface Specifications for project Sayyara. Sayyara is a progressive web application (PWA) which will act as a single platform for independent auto repair shops and vehicle owners. This platform will allow independent auto repair shops and vehicle owners to interact in a more efficient and effective manner. Vehicle owners can search for auto repair shops and services; request quotes for service; book, view, and manage service appointments. On the application, auto repair shop owners will be able to manage a list of employees; manage a list of service types and corresponding service appointment availabilities; manage store information such as location, hours of operation, and contact information. Auto repair shop owners and employees will be able to manage quotes, service appointments, and work orders from a single application. The MIS will detail specifications for the project described above.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at https://github.com/arkinmodi/project-sayyara/.

4 Notation

[You should describe your notation. You can use what is below as a starting point. —SS]

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol := is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1|c_2 \Rightarrow r_2|...|c_n \Rightarrow r_n)$.

The following table summarizes the primitive data types used by Sayyara.

Data Type	Notation	Description
character	char	a single symbol or digit
integer	$\mathbb{Z}$	a number without a fractional component in (-∞, ∞)
natural number	$\mathbb{N}$	a number without a fractional component in [1, ∞)
real	$\mathbb{R}$	any number in (-∞, ∞)

The specification of Sayyara uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, Sayyara uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

5 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	**Level 2**
Hardware-Hiding Module	
Behaviour-Hiding Module	Users Module Quotes Module Appointments Module Work Orders Module Employee Management Module Services Module Shop Module
Software Decision Module	Database Driver Module

Table 2: Module Hierarchy

6 MIS of Database Driver Module

6.1 Module

schema.prisma

6.2 Uses

None

6.3 Syntax

6.3.1 Exported Constants

None

6.3.2 Exported Types

Employee

Output Name	Output Type	Description
id	String	ID of Employee
create_time	String	Create Time of Employee Account
update_time	String	Update Time of Employee Account
first_name	String	First Name of Employee
last_name	String	Last Name of Employee
phone_number	String	Phone Number of Employee
email	String	Email of Employee
password	String	Password of Employee's Account
type	String	Type of User
shop	Shop	Shop which Employee is registered under
appointments	List<Appointment>	Appointments assigned to the Employee

Customer

Output Name	Output Type	Description
id	String	ID of Customer
create_time	String	Create Time of Customer Account
update_time	String	Update Time of Customer Account
first_name	String	First Name of Customer
last_name	String	Last Name of Customer
phone_number	String	Phone Number of Customer
email	String	Email of Customer
password	String	Password of Customer's Account
type	String	Type of User
appointments	List<Appointment>	Appointments assigned to the Customer
chat_messages	List<ChatMessage>	Chat Messages sent by the Customer
quotes	List<Quote>	Quotes initiated by the Customer
vehicles	List<Vehicle>	Vehicles associated with the Customer

Appointment

Output Name	Output Type	Description
id	String	ID of Appointment
create_time	String	Create Time of Appointment
update_time	String	Update Time of Appointment
quote	Optional<Quote>	Associated Quote
work_order	Work Order	Associated Work Order
vehicle	Vehicle	Associated Vehicle
service_type	String	Type of Service
employee	Optional<Employee>	Assigned Employee
customer	Customer	Assigned Customer
status	String	Progress Status of Appointment
start_time	DateTime	Start Time
end_time	DateTime	End Time
shop	Shop	Associated Shop

Quote

Output Name	Output Type	Description
id	String	ID of Quote
create_time	String	Create Time of Quote
update_time	String	Update Time of Quote
customer	Customer	Assigned Customer
shop	Shop	Associated Shop
appointment	Optional<Appointment>	Associated Appointment
chat_messages	List<ChatMessage>	Associated Chat Messages

ChatMessage

Output Name	Output Type	Description
id	String	ID of Chat Message
create_time	String	Create Time of Chat Message
update_time	String	Update Time of Chat Message
message	String	Chat Message
quote	Quote	Associated Quote
customer	Customer	Associated Customer
shop	Shop	Associated Shop

Vehicle

Output Name	Output Type	Description
id	String	ID of the Vehicle
create_time	String	Create Time of the Vehicle
update_time	String	Update Time of the Vehicle
year	$\mathbb{N}$	Model Year of the Vehicle
make	String	Make of the Vehicle
model	String	Model of the Vehicle
vin	String	VIN of the Vehicle
license_plate	String	License Plate of the Vehicle
customer	Customer	Associated Customer
appointments	List<Appointment>	Appointments assigned to the Vehicle

WorkOrder

Output Name	Output Type	Description
id	String	ID of Work Order
create_time	String	Create Time of Work Order
update_time	String	Update Time of Work Order
appointment	Appointment	Associated Appointment
title	String	Title of Work Order
customer	Customer	Assigned Customer
vehicle	Vehicle	Associated Vehicle
employee	Employee	Assigned Employee
body	String	Work Order Details
shop	Shop	Associated Shop

Shop

Output Name	Output Type	Description
id	String	ID of Shop
create_time	String	Create Time of Shop
update_time	String	Update Time of Shop
name	String	Name of Shop
address	String	Address of Shop
city	String	City of Shop
province	String	Province of Shop
postal_code	String	Postal Code of Shop
email	String	Email of Shop
phone_number	String	Phone Number of Shop
hours_of_operation	JSON	Hours of Operation of Shop
appointments	List<Appointment>	Appointments assigned to Shop
employees	List<Employee>	List of Employee assigned to Shop
chat_messages	List<ChatMessage>	List of Chat Messages sent by Shop
quotes	List<Quote>	List of Quotes
services	List<Service>	List of offered Services

6.3.3 Exported Access Programs

None

6.4 Semantics

6.4.1 State Variables

None

6.4.2 Environment Variables

None

6.4.3 Assumptions

None

6.4.4 Access Routine Semantics

None

6.4.5 Local Functions

None

7 MIS of Shop Module

7.1 Module

shopService

7.2 Uses

Database Driver Module

7.3 Syntax

7.3.1 Exported Constants

None

7.3.2 Exported Types

CreateShopType

Output Name	Output Type	Description
name	String	Name of Shop
address	String	Address of shop
city	String	City of Shop
province	String	Province of Shop
postal_code	String	Postal Code of Shop
email	String	Email of Shop
phone_number	String	Phone Number of Shop

UpdateShopType

Output Name	Output Type	Description
name	Optional<String>	Name of Shop
address	Optional<String>	Address of shop
city	Optional<String>	City of Shop
province	Optional<String>	Province of Shop
postal_code	Optional<String>	Postal Code of Shop
hours_of_operation	Optional<JSON>	Hours of Operation of Shop
email	Optional<String>	Email of Shop
phone_number	Optional<String>	Phone Number of Shop

7.3.3 Exported Access Programs

Name	In	Out	Exceptions
createShop	CreateShopType	Shop	
getShopById	String	Shop $\vee$ None	
updateShopById	String, UpdateShopType	Shop	ShopNotFoundException, InvalidTimeException

7.4 Semantics

7.4.1 State Variables

None

7.4.2 Environment Variables

User's Display
Database

7.4.3 Assumptions

None

7.4.4 Access Routine Semantics

createShop(*shop*):

- transition: new Shop(shop), save shop to shop database table
- output: *out* := new Shop(*shop*)

getShopById(*id*):

- output: *out* := A Shop such that it contains the ID, *id*, from the shop database table else None.

updateShopById(*id*, *patch*):

- transition: Update all fields of a Shop, with an ID equal to *id*, with fields in *patch* in shop database table
- output: *out* := $getShopById(id) = \text{Shop} \Rightarrow (\text{Shop} = \text{None}$
  $\Rightarrow$ ShopNotFoundException | Shop $\neq$ None
  $\Rightarrow$ update all fields of Shop with fields in *patch* in shop database table)
- exception: *exc* := $getShopById(id) = None \Rightarrow$ ShopNotFoundException
  | $\exists day \in hours\_of\_operation : day.open\_time > day.close\_time$
  $\Rightarrow$ InvalidTimeException

7.4.5 Local Functions

None

8 MIS of Users Module

8.1 Module

userService

8.2 Uses

Database Driver Module

8.3 Syntax

8.3.1 Exported Constants

None

8.3.2 Exported Types

CreateCustomerType

Output Name	Output Type	Description
email	String	Email of Customer
password	String	Password of Customer Account
first_name	String	First Name of Customer
last_name	String	Last Name of Customer
phone_number	String	Phone Number of Customer
vehicle.year	$\mathbb{N}$	Model Year of Customer's Vehicle
vehicle.make	String	Make of Customer's Vehicle
vehicle.model	String	Model of Customer's Vehicle
vehicle.vin	String	VIN of Customer's Vehicle
vehicle.license_plate	String	License Plate of Customer's Vehicle

CreateEmployeeType

Output Name	Output Type	Description
email	String	Email of Employee
password	String	Password of Employee's Account
first_name	String	First Name of Employee
last_name	String	Last Name of Employee
phone_number	String	Phone Number of Employee
shop_id	String	ID of Shop that Employee Works For

CreateShopOwnerType

Output Name	Output Type	Description
email	String	Email of Shop Owner
password	String	Password of Shop Owner's Account
first_name	String	First Name of Shop Owner
last_name	String	Last Name of Shop Owner
phone_number	String	Phone Number of Shop Owner
shop.name	String	Name of Shop owned by Shop Owner
shop.address	String	Address of Shop owned by Shop Owner
shop.city	String	City of Shop owned by Shop Owner
shop.province	String	Province of Shop owned by Shop Owner
shop.email	String	Email of Shop owned by Shop Owner
shop.phone_number	String	Phone Number of Shop owned by Shop Owner

AuthorizeReturnType

Output Name	Output Type	Description
id	String	User ID
firstName	String	First Name of User
lastName	String	Last Name of User
email	String	Email of User
type	String	Type of User

8.3.3 Exported Access Programs

Name	In	Out	Exceptions
createCustomer	CreateCustomerType	Customer	CustomerAlreadyExistsException
createEmployee	CreateEmployeeType	Employee	EmployeeAlreadyExistsException
createShopOwner	CreateShopOwnerType	Employee	ShopOwnerAlreadyExistsException
getUserByEmail	String	Customer ∨ Employee ∨ None	UserNotFoundException
authorize	String, String	AuthorizeReturnType	UnauthorizeException

8.4 Semantics

8.4.1 State Variables

None

8.4.2 Environment Variables

User's Display
Database

8.4.3 Assumptions

None

8.4.4 Access Routine Semantics

createCustomer(*customer*):

- transition: new Customer(customer), save customer to customer database table

- output: out := (exc = CustomerAlreadyExistsException
  ⇒ "User with email address already exists."| $\neg exc$ ⇒ new Customer($customer$))
- exception: $exc := getUserByEmail(customer.email) \neq$ None
  ⇒ CustomerAlreadyExistsException

createEmployee($employee$):

- transition: new Employee(employee), save employee to employee database table
- output: out := (exc = EmployeeAlreadyExistsException
  ⇒ "User with email address already exists."| $\neg exc$ ⇒ new Employee($employee$))
- exception: $exc := getUserByEmail(employee.email) \neq$ None
  ⇒ EmployeeAlreadyExistsException

createShopOwner($shopOwner$):

- transition: new Employee(shopOwner), save shop owner to employee database table
- output: out := (exc = ShopOwnerAlreadyExistsException
  ⇒ "User with email address already exists."| $\neg exc$ ⇒ new Employee($shopOwner$))
- exception: $exc := getUserByEmail(shopOwner.email) \neq$ None
  ⇒ ShopOwnerAlreadyExistsException

getUserByEmail(e):

- output: out := A User such that it contains the email, e, from the customer database table or employees database table, else None.

authorize($email, password$):

- output: $out := getUserByEmail(email)$ = User ⇒
  (User = None ⇒ exc = UserNotFoundException ⇒ "User not found."
  | User.password = $password$ ⇒ AuthorizeReturnType
  | User.password ≠ $password$ ⇒ exc = UnauthorizeException ⇒ "Unauthorized")

8.4.5 Local Functions

hash(s):

- output: out := hashed value of s

9 Bibliography

Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.

Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL http://citeseer.ist.psu.edu/428727.html.

10 Appendix

[Extra information if required —SS]