Module Interface Specification for Sayyara

Team 31
SFWRENG 4G06
Christopher Andrade
Alyssa Tunney
Kai Zhu
Ethan Vince-Budan
Collin Kan
Harsh Gupta

April 6, 2023

Revision History

Date	**Version**	**Notes**
Jan. 18, 2023	Rev 0	Revision 0
April 5, 2023	Rev 1	Revision 1

Table 1: Revision History

Contents

List of Tables

1 Introduction

The following document details the Module Interface Specifications for Sayyara, a progressive web applications that connects customers with automotive shop owners for vehicle service inquiries, appointment scheduling, and quotes.
Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at https://github.com/atunney/pwa-capstone/tree/main/docs.

2 Module Decomposition

In the Module Guide document, due to the nature of this project, modules have been separated in categories frontend and backend hiding modules. Many backend modules for the project have a corresponding frontend module that displays the information in a visual format for the user to view and interact with.

3 MIS of Quotes (Backend)

3.1 Module

Quotes Backend

3.2 Uses

QuoteRequest
Shop
ShopService

3.3 Syntax

3.3.1 Exported Constants

PRIORITIES = ('low', 'medium', 'high')
BOOKING_STATUS = ('pending', 'canceled', 'booked')

3.3.2 Exported Access Programs

Name	In	Out	Exceptions
getQuotes	shop: Shop	Quote	NotFound
updateQuote	id: TextField	-	InvalidFields
	shop: Shop		
	quote_request: QuoteRequest		
	services: ShopService		
	status: TextField		
	priority: TextField		
	message: TextField		
createQuote	id: TextField	-	-
	quote_request: QuoteRequest		

3.4 Semantics

3.4.1 State Variables

shop: Shop
quote_request: Array: QuoteRequest
services: ShopService
status: TextField
priority: TextField
message: TextField

3.4.2 Environment Variables

Quotes: Array: Quotes QuoteRequests: Array: QuoteRequest

3.4.3 Assumptions

- In order for a shop to modify and send updated quotes, a customer needs to send a quote request to said shop.
- All fields in a Quote will originally be filled out by a customer.

3.4.4 Access Routine Semantics

getQuotes(shop: Shop):

- transition: none
- output: Array: Quote in Quotes such that Quote.shop.id == shop.id
- exception: NotFound: no quotes with associated shop

updateQuote(id: TextField, shop: Shop, quote_request: QuoteRequest, services: ShopService, status: TextField, priority: TextField, message: TextField):

- transition: for Quote in Quotes such that Quote.id == id: Quote.shop = shop, Quote.quote_request = quote_request, Quote.services = services, Quote.status = status, Quote.priority = priority, Quote.message = message
- output: none
- exception: InvalidFields: one or more of inputs is in invalid format

createQuote(id: TextField, quote_request: QuoteRequest):

- transition: Quotes = Quotes $\cup$ Quote(id = id, quote_request = quote_request)
- output: none
- exception: none

3.4.5 Local Functions

N/A

4 MIS of QuoteRequest (Backend)

4.1 Module

QuoteRequest Backend

4.2 Uses

User
Vehicle

4.3 Syntax

4.3.1 Exported Constants

N/A

4.3.2 Exported Access Programs

Name	In	Out	Exceptions
getQuoteRequests	customer: Customer	QuoteRequest	NotFound
getVehicles	customer: Customer	Vehicle	NotFound
updateQuoteRequest	part_preference: TextField customer: Customer description: TextField images: TextField vehicle: Vehicle availability: TextField	-	InvalidFields
deleteQuoteRequest	id: TextField	-	NotFound
createQuoteRequest	part_preference: TextField customer: Customer description: TextField vehicle: Vehicle availability: TextField	-	InvalidFields

4.4 Semantics

4.4.1 State Variables

id: TextField
part_preference: TextField
description: TextField
images: TextField
vehicle: Vehicle

created_at: DateTimeField
availability: TextField

4.4.2 Environment Variables

QuoteRequests: Array: QuoteRequest

4.4.3 Assumptions

N/A

4.4.4 Access Routine Semantics

getQuoteRequests(customer: Customer):

- transition: none
- output: Array: QuoteRequest in QuoteRequests such that QuoteRequest.customer.id == customer.id
- exception: NotFound: no quote requests with associated customer

getVehicles(customer: Customer)

- transition: none
- output: customer.vehicles
- exception: NotFound: no vehicles with associated customer

updateQuoteRequest(id: TextField, part_preference: TextField, description: TextField, images: TextField, vehicle: Vehicle, availability: TextField)

- transition: for QuoteRequest such that QuoteRequest.id = id: QuoteRequest.part_preference = part_preference, QuoteRequest.description = description, QuoteRequest.images = images, QuoteRequest.vehicle = vehicle, QuoteRequest.availability = availability
- output: none
- exception: InvalidFields: one or more of inputs is in invalid format

deleteQuoteRequest(id: TextField)

- transition: delete QuoteRequest from QuoteRequests where QuoteRequest.id == id
- output:
- exception: NotFound: no QuoteRequests with associated id

createQuoteRequest(id: TextField, part_preference: TextField, description: TextField, vehicle: Vehicle, availability: TextField)

- transition: QuoteRequests = QuoteRequests $\cup$ QuoteRequest(id = id, part_preference = part_preference, description = description, vehicle = vehicle, availability = availability)
- output: none
- exception: InvalidFields: one or more of inputs is in invalid format

4.4.5 Local Functions

N/A

5 MIS of Appointments (Backend)

5.1 Module

Appointment Backend

5.2 Uses

Customer
Shop
Quote
EmployeeReservation

5.3 Syntax

5.3.1 Exported Constants

PRIORITIES = ('low', 'medium', 'high')
BOOKING_STATUS = ('pending', 'canceled', 'booked')

5.3.2 Exported Access Programs

Name	In	Out	Exceptions
createAppointment	data: Appointment	Appointment	-
updateAppointment	appointment: Appointment title: TextField description: TextField time_estimate: DurationField price_estimate: DecimalField quote: Quote message: TextField	-	-
setStatus	appointment: Appointment status: TextField	-	InvalidFields

5.4 Semantics

5.4.1 State Variables

customer: Customer
shop: Shop
title: TextField
description: TextField
time_estimate: DurationField
price_estimate: DecimalField

quote: Quote
message: TextField

5.4.2 Environment Variables

appointments: Array: Appointment

5.4.3 Assumptions

- The customer must fill out all relevant fields in the appointment in order to book it.
- An appointment is only valid when a valid quote is supplied along with it.

5.4.4 Access Routine Semantics

createAppointment(customer: Customer, shop: Shop, title: TextField, description: TextField, time_estimate: DurationField, price_estimate: DecimalField, quote: Quote, message: TextField):

- transition: Appointments = Appointments ∪ Appointment(customer = customer, shop = shop, title = title, description = description, time_estimate = time_estimate, price_estimate = price_estimate, quote = quote, message = message)
- output: Appointment
- exception: none

updateAppointment(appointment: Appointment, title: TextField, description: TextField, time_estimate: DurationField, price_estimate: DecimalField, quote: Quote, message: TextField):

- transition: for Appointment in Appointments such that Appointment.id == appointment.id: Appointment.title = title, Appointment.description = description, Appointment.time_estimate = time_estimate, Appointment.price_estimate = price_estimate, Appointment.quote = quote, Appointment.message = message
- output: none
- exception: none

setStatus(appointment: Appointment, status: TextField):

- transition: for Appointment in Appointments such that Appointment.id == appointment.id, Appointment.status = status
- output: none
- exception: InvalidFields: status is invalid

5.4.5 Local Functions

N/A

6 MIS of Shop Quotes (Frontend)

6.1 Module

Quotes Frontend

6.2 Uses

Quotes Backend
QuoteRequest Backend
Shop
ShopService

6.3 Syntax

6.3.1 Exported Constants

N/A

6.3.2 Exported Access Programs

Name	In	Out	Exceptions
getComponent	-	QuotesShop	-

6.4 Semantics

6.4.1 State Variables

shopData: Shop
quoteRequests: Array: QuoteRequest
services: ShopService
status: TextField
priority: TextField
message: TextField

6.4.2 Environment Variables

N/A

6.4.3 Assumptions

6.4.4 Access Routine Semantics

getComponent():

- transition: none
- output: QuotesShop
- exception: none

6.4.5 Local Functions

setShopData(shop: Shop)
setQuoteRequests(quoteRequests: Array: QuoteRequest)
setServices(services: Array: ShopService)
setStatus(status: TextField)
setPriority(priority: TextField)
setMessage(message: TextField)

7 MIS of Customer QuoteRequests (Frontend)

7.1 Module

QuoteRequests Frontend

7.2 Uses

QuoteRequests Backend
User
Vehicle
Shop

7.3 Syntax

7.3.1 Exported Constants

N/A

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
getComponent	-	QuotesCustomer	-

7.4 Semantics

7.4.1 State Variables

shopData: Array: Shop
userData: Array: User
partPreference: TextField
vehicleData: Array: VehicleMakes
vehicleModels: Array: VehicleModels
vehicleYears: Array: TextField
description: TextField
availability: TextField

7.4.2 Environment Variables

N/A

7.4.3 Assumptions

N/A

7.4.4 Access Routine Semantics

getComponent():

- transition: none
- output: QuotesCustomer
- exception: none

7.4.5 Local Functions

setShopData(shop: Shop)
setUserData(user: User)
setPartPreference(preference: TextField)
setVehicleData(vehicleMakes: Vehicle)
setVehicleModels(vehicleModels: Vehicle)
setVehicleYears(years: Array: TextField)
setDescription(description: TextField)
setAvailability(availability: TextField)

8 MIS of Shop (Backend)

8.1 Module

Shop Backend

8.2 Uses

User
ShopService

8.3 Syntax

8.3.1 Exported Constants

N/A

8.3.2 Exported Access Programs

Name	In	Out	Exceptions
getShop	id : TextField	Shop	NotFound
createShop	name : TextField address: TextField email: TextField phone: TextField description: TextField owner: User	-	InvalidFields
updateShop	id: TextField name : TextField address: TextField email: TextField phone: TextField description: TextField services: Array: ShopService employees: Array: User	-	InvalidFields
deleteShop	id : TextField	-	NotFound

8.4 Semantics

8.4.1 State Variables

id: TextField
owner: User

employees: Array: User
services: Array: ShopService
address: TextField
description: TextField
email: TextField
hours: TextField
name: TextField
phone: TextField

8.4.2 Environment Variables

Shops: Array: Shop

8.4.3 Assumptions

N/A

8.4.4 Access Routine Semantics

getShop(id: TextField):

- transition: none
- output: Shop in Shops such that Shop.id == id
- exception: NotFound: for every Shop in Shops: Shop.id != id

createShop(name : TextField, address: TextField, email: TextField, phone: TextField, description: TextField, owner: User):

- transition: Shops = Shops ∪ Shop(name = name, address = address, email = email, phone = phone, description = description, owner = owner)
- output: none
- exception: InvalidFields: one or more of inputs is in invalid format

updateShop(id: TextField, name : TextField, address: TextField, email: TextField, phone: TextField, description: TextField, services: Array<ShopService>, employees: Array<User>):

- transition: for Shop in Shops such that Shop.id == id: Shop.name = name, Shop.address = address, Shop.email = email, Shop.phone = phone, Shop.description = description, Shop.services = services, Shop.employees = employees
- output: none
- exception: InvalidFields: one or more of inputs is in invalid format

deleteShop(id: TextField):

- transition: delete Shop from Shops where Shop.id == id
- output: none
- exception: NotFound: for every ShopService in ShopServices: ShopService.id != id

8.4.5 Local Functions

N/A

9 MIS of ShopService (Backend)

9.1 Module

ShopService Backend

9.2 Uses

Service

9.3 Syntax

9.3.1 Exported Constants

N/A

9.3.2 Exported Access Programs

Name	In	Out	Exceptions
getShopService	id : TextField	ShopService	NotFound
createShopService	service : Service est_time: TextField price: TextField	-	InvalidFields
updateShopService	id: TextField service : Service est_time: TextField price: TextField	-	InvalidFields
deleteShopService	id : TextField	-	NotFound

9.4 Semantics

9.4.1 State Variables

id: TextField
service: Service
est_time : TextField
price: TextField

9.4.2 Environment Variables

ShopServices: Array: ShopService

9.4.3 Assumptions

N/A

9.4.4 Access Routine Semantics

getShopService(id: TextField):

- transition: none
- output: ShopService in ShopServices such that ShopService.id == id
- exception: NotFound: for every ShopService in ShopServices: ShopService.id != id

createShopService(service : Service, est_time: TextField, price: TextField):

- transition: ShopServices = ShopServices ∪ ShopService(service = service, est_time = est_time, price = price)
- output: none
- exception: InvalidFields: one or more of inputs is in invalid format

updateShopService(id: TextField, service : Service, est_time: TextField, price: TextField):

- transition: for ShopService in ShopServices such that ShopService.id == id: ShopService.service = service, Shop.est_time = est_time, Shop.price = price
- output: none
- exception: InvalidFields: one or more of inputs is in invalid format

deleteShop(id: TextField):

- transition: delete ShopService from ShopServices where ShopService.id == id
- output: none
- exception: NotFound: for every ShopService in ShopServices: Shop.id != id

9.4.5 Local Functions

N/A

10 MIS of Service (Backend)

10.1 Module

Service Backend

10.2 Uses

N/A

10.3 Syntax

10.3.1 Exported Constants

N/A

10.3.2 Exported Access Programs

Name	In	Out	Exceptions
getService	id : TextField	Service	NotFound
createService	description : TextField name: TextField	-	InvalidFields
updateService	id: TextField description : TextField name: TextField	-	InvalidFields
deleteService	id : TextField	-	NotFound

10.4 Semantics

10.4.1 State Variables

id: TextField
description: TextField
name: TextField

10.4.2 Environment Variables

Services: Array: Service

10.4.3 Assumptions

N/A

10.4.4 Access Routine Semantics

getService(id: TextField):

- transition: none
- output: Service in Services such that Service.id == id
- exception: NotFound: for every Service in Services: Service.id != id

createService(description : TextField, name: TextField):

- transition: Services = Services $\cup$ Service(description = description, name = name)
- output: none
- exception: InvalidFields: one or more of inputs is in invalid format

updateService(id: TextField, description : TextField, name: TextField):

- transition: for Service in Services such that Service.id == id: Service.description = description, Shop.name = name
- output: none
- exception: InvalidFields: one or more of inputs is in invalid format

deleteShop(id: TextField):

- transition: delete Service from Services where Service.id == id
- output: none
- exception: NotFound: for every Service in Services: Service.id != id

10.4.5 Local Functions

N/A

11 MIS of ShopCreate (Frontend)

11.1 Module

ShopCreate Frontend

11.2 Uses

Shop (Backend)

11.3 Syntax

11.3.1 Exported Constants

N/A

11.3.2 Exported Access Programs

Name	In	Out	Exceptions
getComponent	-	ShopCreate	-

11.4 Semantics

11.4.1 State Variables

shopName: TextField
shopAddress: TextField
shopEmail: TextField
shopPhone: TextField
shopDescription: TextField
isShopCreated: boolean
isShopCreating: boolean
isShopCreationError: boolean

11.4.2 Environment Variables

N/A

11.4.3 Assumptions

N/A

11.4.4 Access Routine Semantics

getComponent():

- transition: none
- output: ShopCreate
- exception: none

11.4.5 Local Functions

setShopName(name : TextField): ShopCreate.shopName = name
setShopAddress(address : TextField): ShopCreate.shopAddress = address
setShopEmail(email : TextField): ShopCreate.shopEmail = email
setShopPhone(phone : TextField): ShopCreate.shopPhone = phone
setShopDescription(description : TextField): ShopCreate.shopDescription = description
setIsShopCreated(isCreated : boolean): ShopCreate.isShopCreated = isCreated
setIsShopCreating(isCreating : boolean): ShopCreate.isShopCreating = isCreating

12 MIS of ShopLookup (Frontend)

12.1 Module

ShopLookup Frontend

12.2 Uses

Shop
ShopService
Service

12.3 Syntax

12.3.1 Exported Constants

N/A

12.3.2 Exported Access Programs

Name	In	Out	Exceptions
getComponent	-	ShopLookup	-

12.4 Semantics

12.4.1 State Variables

allShops: Array: Shop
queryShops: Array: Shop
searchQuery: TextField

12.4.2 Environment Variables

N/A

12.4.3 Assumptions

N/A

12.4.4 Access Routine Semantics

getComponent():

- transition: none

- output: ShopLookup
- exception: none

12.4.5 Local Functions

getAllShops()
setAllShops(shops: Array: Shop)
setSearchQuery(query : TextField)
getQueryShops(query : TextField)
setQueryShops(results : Array: Shop)

13 MIS of Profile (Backend)

13.1 Module

Profile Backend

13.2 Uses

User

13.3 Syntax

13.3.1 Exported Constants

ACCOUNT_TYPE: Tuple: (”ShopEmployee”, ”ShopOwner”, ”Customer”)

13.3.2 Exported Access Programs

Name	In	Out	Exceptions
getProfile	pk: DecimalField	Profile	NotFound
updateProfile	pk: DecimalField first_name: TextField last_name: TextField email: TextField phone: TextField role: ACCOUNT_TYPE	Profile	NotFound InvalidFields
createProfile	username: TextField password: TextField first_name: TextField last_name: TextField email: TextField phone: TextField role: ACCOUNT_TYPE	-	UserExists InvalidFields
resetRequest	id: TextField email: TextField	token: TextField	InvalidFields
resetValidate	id: TextField token: TextField	valid: BooleanField	-
resetConfirm	id: TextField token: TextField password: TextField	-	InvalidToken InvalidFields

13.4 Semantics

13.4.1 State Variables

role: ACCOUNT_TYPE
phone: TextField
user: User

13.4.2 Environment Variables

Profiles: Array: Profile

13.4.3 Assumptions

N/A

13.4.4 Access Routine Semantics

getProfile(pk: DecimalField):

- transition: none
- output: Profile in Profiles such that Profile.user.pk == pk
- exception: NotFound - pk does not point to an existing user

updateProfile(pk: DecimalField, first_name: TextField, last_name: TextField, email: TextField, phone: TextField, role: ACCOUNT_TYPE):

- transition: for Profile in Profiles such that Profile.user.pk == pk: Profile.phone = phone, profile.user.first_name = first_name, profile.user.last_name=last_name, profile.user.email = email, profile.role = role
- output: none
- exception: NotFound - pk does not point to an existing user; InvalidFields - any input is empty or contains invalid formatting (phone, email, role)

createProfile(username: TextField, password: TextField, first_name: TextField, last_name: TextField, email: TextField, phone: TextField, role: ACCOUNT_TYPE):

- transition: Profiles = Profiles ∪ Profile(phone = phone, role = role, User(username = username, password = password, email = email, first_name = first_name, last_name = last_name))
- output: none
- exception: UserExists - users[pk] already exists; InvalidFields - any input is empty or contains invalid formatting (phone, email, role)

resetRequest(id: DecimalField, email: TextField):

- transition: for Profile in Profiles such that Profile.user.pk == id: Profile.user.token = GenerateToken()
- output: for Profile in Profiles such that Profile.user.pk == id: SendResetEmail(email = email, token = Profile.user.token, id = Profile.user.id), return Token
- exception: InvalidFields - email input is incorrectly formatted

resetValidate(id: DecimalField, token: TextField):

- transition: none
- output: for Profile in Profiles such that Profile.user.pk == id: return Profile.user.token == token
- exception: N/A

resetConfirm(id: DecimalField, token: TextField, password: TextField):

- transition: for Profile in Profiles such that Profile.user.pk == id and Profile.user.token == token: Profile.user.password = password
- output: N/A
- exception: InvalidToken - uid and token cannot be validated; InvalidFields - password is invalidly formatted (password.length < 8 or contains invalid characters)

13.4.5 Local Functions

generateToken(): A function that generates and returns a new random Token for verification purposes
SendResetEmail(email: TextField, token: TextField, id: DecimalField): A function that creates a unique URL to a reset password page and emails the URL to the given email

14 MIS of AppContext (Frontend)

14.1 Module

AppContext Frontend

14.2 Uses

N/A

14.3 Syntax

14.3.1 Exported Constants

AppWrapper: React.FunctionalComponent

14.3.2 Exported Access Programs

Name	**In**	**Out**	**Exceptions**
setUser	user: Dictionary	-	-
setTokens	tokens: Dictionary	-	-

14.4 Semantics

14.4.1 State Variables

user: Dictionary: {username: TextField, email: TextField, role: TextField, first_name: TextField, last_name: TextField, phone: TextField}
tokens: Dictionary: {accessToken: TextField, refreshToken: TextField, authStatus: TextField}

14.4.2 Environment Variables

LocalStorage : Dictionary : any

14.4.3 Assumptions

Assume that the Dictionary type of user and tokens corresponds to the detailed Dictionary type shown in the above State Variables section (to avoid reptition and improve readability)

14.4.4 Access Routine Semantics

setUser(user: Dictionary):

- transition: AppContext.user = user

- output: none
- exception: none

setTokens(tokens: Dictionary):

- transition: AppContext.tokens = tokens
- output: none
- exception: none

14.4.5 Local Functions

N/A

15 MIS of Authentication (Frontend)

15.1 Module

Authentication Frontend

15.2 Uses

AppContext

15.3 Syntax

15.3.1 Exported Constants

N/A

15.3.2 Exported Access Programs

Name	In	Out	Exceptions
getComponent	-	Authentication	-

15.4 Semantics

15.4.1 State Variables

N/A

15.4.2 Environment Variables

N/A

15.4.3 Assumptions

15.4.4 Access Routine Semantics

getComponent():

- transition: none
- output: Authentication
- exception: none

15.4.5 Local Functions

register(): A function that re-renders this page with account registration instructions
login(message: TextField, type: TextField): A function that re-renders this page with a message
forgotten(): A function that re-renders this page with a form to enter an email to send a reset password link to that email
reset(uid: TextField, token: TextField): A function that re-renders this page with a form to reset the password if the given uid and token are valid from the backend
fetchUserProfile(context: AppContext, token: TextField): A function that updates the context with the user's information from the backend (using the token)

16 MIS of Appointments (Backend)

16.1 Module

Appointments Backend

16.2 Uses

Customer
Shop
Quote
EmployeeReservation

16.3 Syntax

16.3.1 Exported Constants

N/A

16.3.2 Exported Access Programs

Name	In	Out	Exceptions
getAppointments	customer: Customer shop: Shop	List[Appointment]	ShopNotFound CustomerNotFound
createAppointment	formData:Appointment	-	ShopNotFound CustomerNotFound

16.4 Semantics

16.4.1 State Variables

- customer: Customer
- shop: Shop
- created_at: DateTimeField
- title: TextField
- description: TextField
- time_estimate: DurationField
- price_estimate: DecimalField

- quote: Quote
- message: TextField
- reservation: EmployeeReservation

16.4.2 Environment Variables

N/A

16.4.3 Assumptions

N/A

16.4.4 Access Routine Semantics

getAppointment(customer: Customer, shop:Shop):

- transition: N/A
- output: List[Appointment] where appointment.customer = customer OR appointment.shop = shop
- exception: ShopNotFound, CustomerNotFound

createAppointment(appointment: Appointment):

- transition: none
- output: List[Appointment]
- exception: ShopNotFound, CustomerNotFound

16.4.5 Local Functions

validateAppointment(instance:Appointment)

17 MIS for VehicleInfo (Backend)

17.1 Uses

Vehicle, VehicleSerializer

17.2 Syntax

17.2.1 Exported Constants

N/A

17.2.2 Exported Access Programs

Name	In	Out	Exceptions
createVehicle	make : TextField model : TextField year : DecimalField		
getVehicle	id : DecimalField	vehicle : Vehicle	VehicleNotFound
putVehicle	id : DecimalField make : TextField model : TextField year : DecimalField		VehicleNotFound
deleteVehicle	id : DecimalField		VehicleNotFound

17.3 Semantics

17.3.1 State Variables

- VehicleList: set of Vehicle

17.3.2 Environment Variables

N/A

17.3.3 Assumptions

- input year is a valid year

17.3.4 Access Routine Semantics

createVehicle(make,model,year):

- transition: VehicleList.append(new Vehicle(make, model, year)) getVehicle(id):

- output: out := VehicleList(id)
- exception: id $\notin$ VehicleList $\Rightarrow$ VehicleNotFound

putVehicle(id, make, model, year):

- transition: VehicleList(id) := new Vehicle(make, model, year)
- exception: id $\notin$ VehicleList $\Rightarrow$ VehicleNotFound

deleteVehicle(id):

- transition: VehicleList(id) := *null*
- exception: id $\notin$ VehicleList $\Rightarrow$ VehicleNotFound

17.3.5 Local Functions

N/A

18 MIS for CustomerProfile (Frontend)

18.1 Uses

Auth, AppContext, Profile, Vehicles

18.2 Syntax

18.2.1 Exported Constants

N/A

18.2.2 Exported Access Programs

Name	**In**	**Out**	**Exceptions**
setUserInfo	pk : int		
updateUserInfo	username: TextField		
	email: TextField		
	role: TextField		
	first_name: TextField		
	last_name: TextField		
	phone: TextField		

18.3 Semantics

18.3.1 State Variables

- userInfo : User

18.3.2 Environment Variables

N/A

18.3.3 Assumptions

- Authentication status and user access tokens are available through the Auth module

18.3.4 Access Routine Semantics

setUserInfo(pk):

- transition: userInfo := Profile.get(pk)

updateUserInfo(username, email, TextField, role, first_name, last_name, phone):

- transition: userInfo := new User(username, email, role, first_name, last_name, phone)
  Profile.update(username, email, role, first_name, last_name, phone)

18.3.5 Local Functions

N/A

19 MIS for ShopOwnerProfile (Frontend)

19.1 Uses

Auth, AppContext, User, Profile, Shop, ShopServices

19.2 Syntax

19.2.1 Exported Constants

N/A

19.2.2 Exported Access Programs

Name	In	Out	Exceptions
setUserInfo	pk : DecimalField		
updateUserInfo	username : TextField		
	email : TextField		
	role : TextField		
	first_name : TextField		
	last_name : TextField		
	phone : TextField		
setShopInfo	id : TextField		
updateShopInfo	id : TextField		
	name : TextField		
	address : TextField		
	email : TextField		
	phone : TextField		
	desc : TextField		
setServicesInfo	ids : set of TextField		
updateServicesInfo	id : TextField		
	service : Service		
	est_time : TextField		
	price : TextField		

19.3 Semantics

19.3.1 State Variables

- userInfo : User
- shopInfo : Shop
- services : set of Services

19.3.2 Environment Variables

N/A

19.3.3 Assumptions

N/A

19.3.4 Access Routine Semantics

setUserInfo(pk):

- transition: userInfo := Profile.get(pk)

updateUserInfo(username, email, TextField, role, first_name, last_name, phone):

- transition: userInfo := new User(username, email, role, first_name, last_name, phone)
  Profile.update(username, email, role, first_name, last_name, phone)

setShopInfo(id):

- transition: shopInfo := Shop.getShop(id)

updateShopInfo(id, name, address, email, phone, desc):

- transition: shopInfo := new Shop(name, address, email, phone, desc)
  Shop.updateShop(id, name, address, email, phone, desc)

setServicesInfo(ids):

- transition: $\forall i \in ids$: services.append(ShopService.getShopService(id))

updateServicesInfo(id, service, est_time, price):

- transition: services(id) := new Service(service)
  ShopService.updateShopService(id, service, est_time, price)

19.3.5 Local Functions

N/A

20 MIS for EmployeeProfile (Frontend)

20.1 Uses

Auth, AppContext, User, Profile, EmployeeAvailability

20.2 Syntax

20.2.1 Exported Constants

N/A

20.2.2 Exported Access Programs

Name	In	Out	Exceptions
setUserInfo	pk : DecimalField		
setAvailability			
updateAvailability			

20.3 Semantics

20.3.1 State Variables

- userInfo : User

20.3.2 Environment Variables

N/A

20.3.3 Assumptions

N/A

20.3.4 Access Routine Semantics

N/A setUserInfo(pk):

- transition: userInfo := Profile.get(pk)

setAvailability():
updateAvailability():

20.3.5 Local Functions

N/A

21 MIS for Homepages (Frontend)

21.1 Uses

Auth, AppContext, User, Profile, Appointments

21.2 Syntax

21.2.1 Exported Constants

N/A

21.2.2 Exported Access Programs

Name	In	Out	Exceptions
getUserInfo	id : DecimalField		
getAppointments	customer : User		

21.3 Semantics

21.3.1 State Variables

- userInfo : User
- userAppointments : set of Appointment

21.3.2 Environment Variables

N/A

21.3.3 Assumptions

N/A

21.3.4 Access Routine Semantics

getUserInfo(id):

- transition: userInfo := Profile.get(id)

getAppointments(customer)

- transition: $\forall s \in Shop(id)$: userAppointments.append(s)

21.3.5 Local Functions

N/A