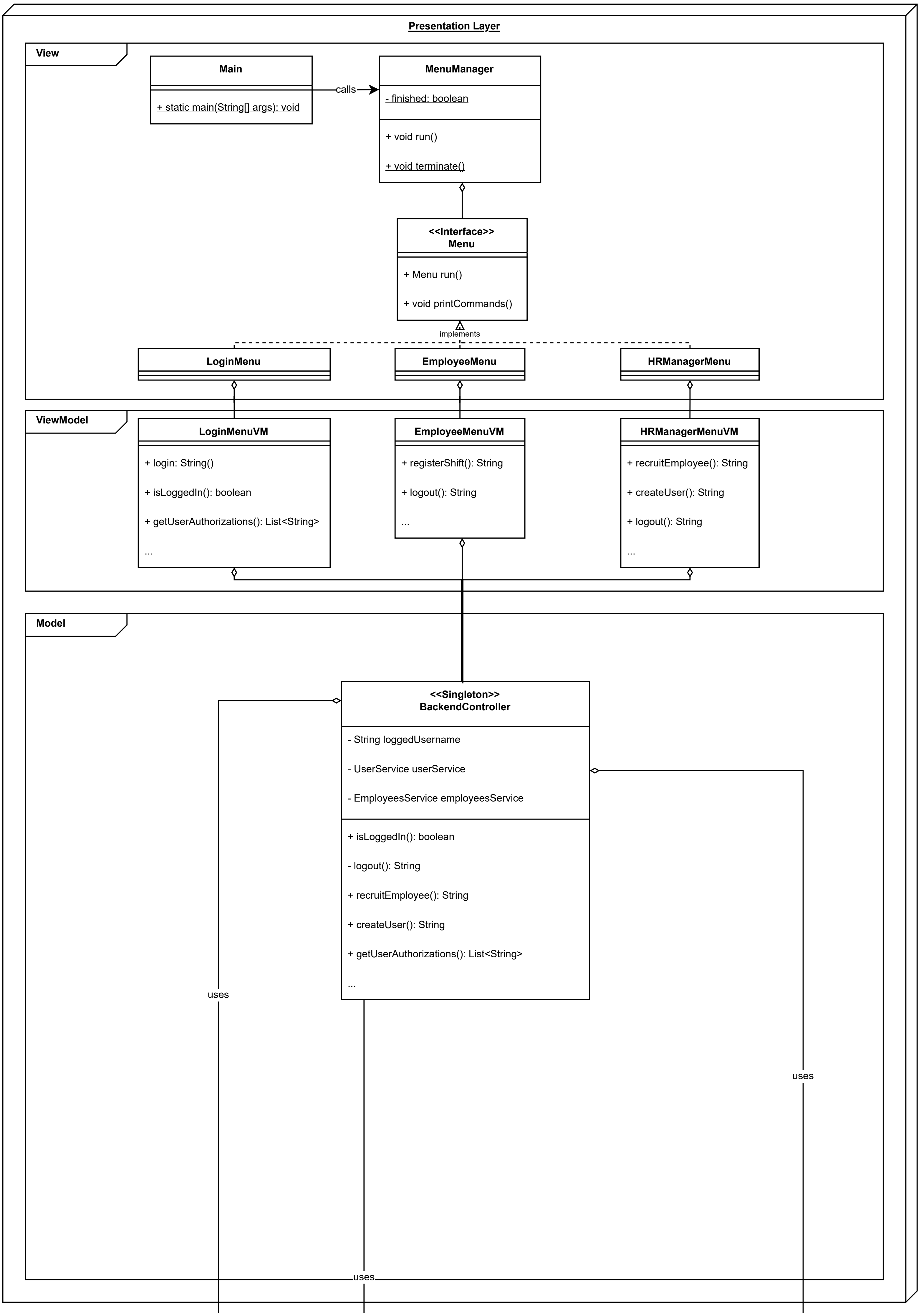
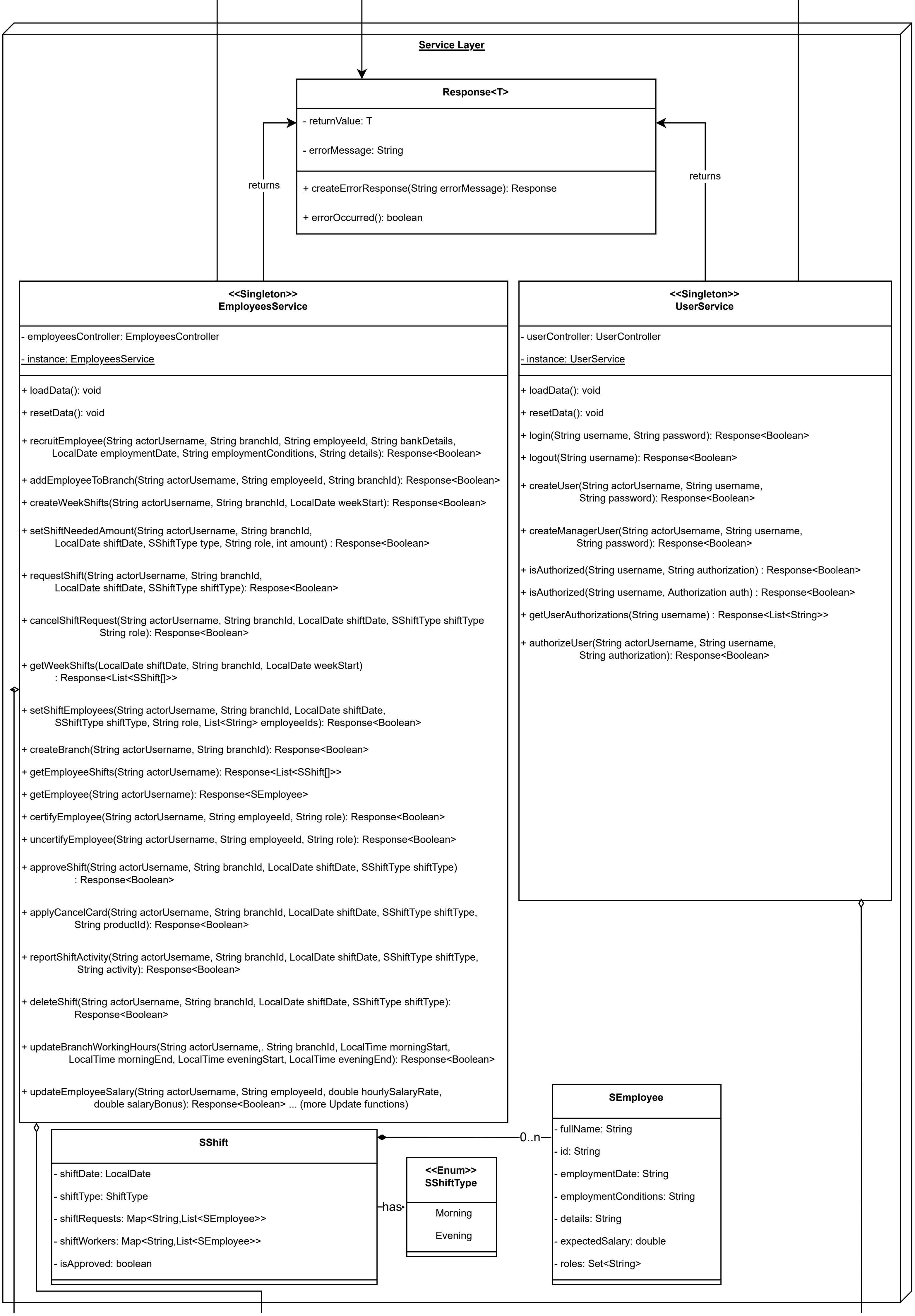
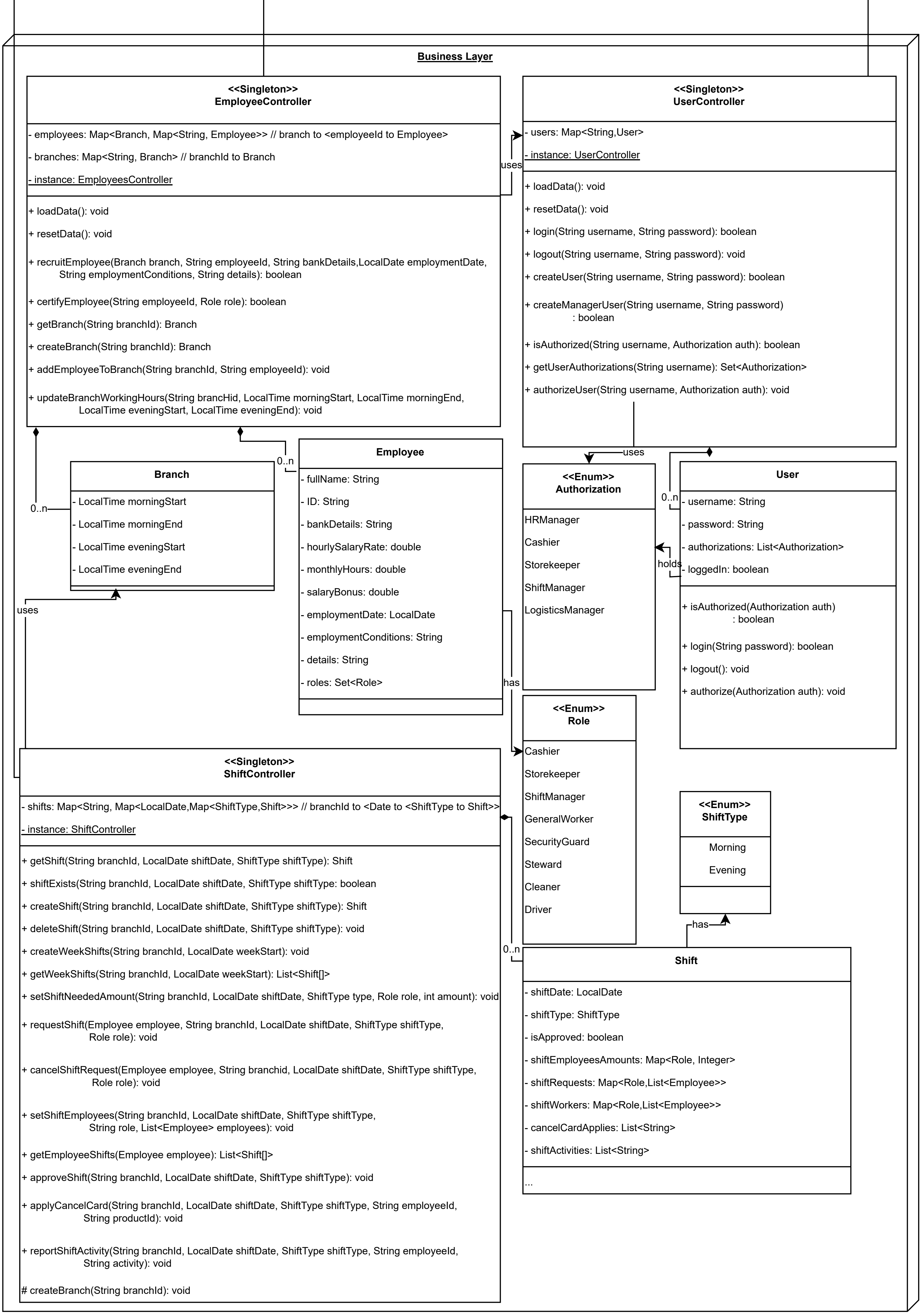
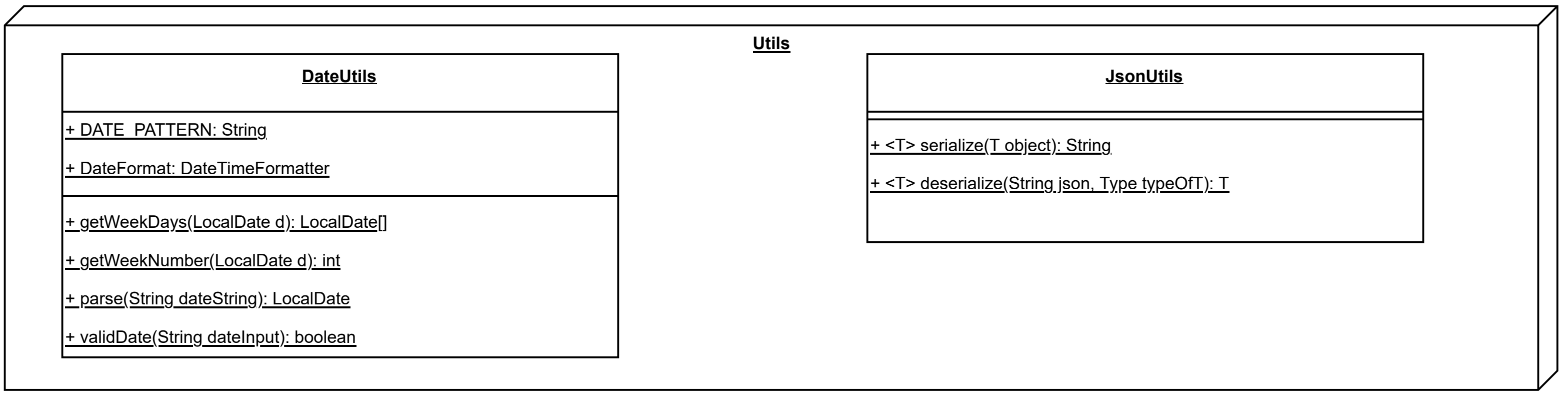

Presentation Layer
View
Main
+ static main(String[] args): void
calls
MenuManager
- finished: boolean
+ void run()
+ void terminate()
<<Interface>>
Menu
+ Menu run()
+ void printCommands()
implements
LoginMenu
EmployeeMenu
HRManagerMenu
ViewModel
LoginMenuVM
+ login: String()
+ isLoggedIn(): boolean
+ getUserAuthorizations(): List<String>
...
EmployeeMenuVM
+ registerShift(): String
+ logout(): String
...
HRManagerMenuVM
+ recruitEmployee(): String
+ createUser(): String
+ logout(): String
...
Model
<<Singleton>>
BackendController
- String loggedUsername
- UserService userService
- EmployeesService employeesService
+ isLoggedIn(): boolean
- logout(): String
+ recruitEmployee(): String
+ createUser(): String
+ getUserAuthorizations(): List<String>
...
uses
uses
uses


Service Layer
Response<T>
- returnValue: T
- errorMessage: String
+ createErrorResponse(String errorMessage): Response
+ errorOccurred(): boolean
returns
returns
<<Singleton>>
EmployeesService
- employeesController: EmployeesController
- instance: EmployeesService
+ loadData(): void
+ resetData(): void
+ recruitEmployee(String actorUsername, String branchId, String employeeId, String bankDetails, LocalDate employmentDate, String employmentConditions, String details): Response<Boolean>
+ addEmployeeToBranch(String actorUsername, String employeeId, String branchId): Response<Boolean>
+ createWeekShifts(String actorUsername, String branchId, LocalDate weekStart): Response<Boolean>
+ setShiftNeededAmount(String actorUsername, String branchId, LocalDate shiftDate, SShiftType type, String role, int amount) : Response<Boolean>
+ requestShift(String actorUsername, String branchId, LocalDate shiftDate, SShiftType shiftType): Respose<Boolean>
+ cancelShiftRequest(String actorUsername, String branchId, LocalDate shiftDate, SShiftType shiftType String role): Response<Boolean>
+ getWeekShifts(LocalDate shiftDate, String branchId, LocalDate weekStart) : Response<List<SShift[]>>
+ setShiftEmployees(String actorUsername, String branchId, LocalDate shiftDate, SShiftType shiftType, String role, List<String> employeeIds): Response<Boolean>
+ createBranch(String actorUsername, String branchId): Response<Boolean>
+ getEmployeeShifts(String actorUsername): Response<List<SShift[]>>
+ getEmployee(String actorUsername): Response<SEmployee>
+ certifyEmployee(String actorUsername, String employeeId, String role): Response<Boolean>
+ uncertifyEmployee(String actorUsername, String employeeId, String role): Response<Boolean>
+ approveShift(String actorUsername, String branchId, LocalDate shiftDate, SShiftType shiftType) : Response<Boolean>
+ applyCancelCard(String actorUsername, String branchId, LocalDate shiftDate, SShiftType shiftType, String productId): Response<Boolean>
+ reportShiftActivity(String actorUsername, String branchId, LocalDate shiftDate, SShiftType shiftType, String activity): Response<Boolean>
+ deleteShift(String actorUsername, String branchId, LocalDate shiftDate, SShiftType shiftType): Response<Boolean>
+ updateBranchWorkingHours(String actorUsername,. String branchId, LocalTime morningStart, LocalTime morningEnd, LocalTime eveningStart, LocalTime eveningEnd): Response<Boolean>
+ updateEmployeeSalary(String actorUsername, String employeeId, double hourlySalaryRate, double salaryBonus): Response<Boolean> ... (more Update functions)
<<Singleton>>
UserService
- userController: UserController
- instance: UserService
+ loadData(): void
+ resetData(): void
+ login(String username, String password): Response<Boolean>
+ logout(String username): Response<Boolean>
+ createUser(String actorUsername, String username, String password): Response<Boolean>
+ createManagerUser(String actorUsername, String username, String password): Response<Boolean>
+ isAuthorized(String username, String authorization) : Response<Boolean>
+ isAuthorized(String username, Authorization auth) : Response<Boolean>
+ getUserAuthorizations(String username) : Response<List<String>>
+ authorizeUser(String actorUsername, String username, String authorization): Response<Boolean>
SShift
- shiftDate: LocalDate
- shiftType: ShiftType
- shiftRequests: Map<String,List<SEmployee>>
- shiftWorkers: Map<String,List<SEmployee>>
- isApproved: boolean
has
<<Enum>>
SShiftType
Morning
Evening
0..n
SEmployee
- fullName: String
- id: String
- employmentDate: String
- employmentConditions: String
- details: String
- expectedSalary: double
- roles: Set<String>


Business Layer
<<Singleton>>
EmployeeController
- employees: Map<Branch, Map<String, Employee>> // branch to <employeeId to Employee>
- branches: Map<String, Branch> // branchId to Branch
- instance: EmployeesController
+ loadData(): void
+ resetData(): void
+ recruitEmployee(Branch branch, String employeeId, String bankDetails,LocalDate employmentDate, String employmentConditions, String details): boolean
+ certifyEmployee(String employeeId, Role role): boolean
+ getBranch(String branchId): Branch
+ createBranch(String branchId): Branch
+ addEmployeeToBranch(String branchId, String employeeId): void
+ updateBranchWorkingHours(String brancHid, LocalTime morningStart, LocalTime morningEnd, LocalTime eveningStart, LocalTime eveningEnd): void
uses
<<Singleton>>
UserController
- users: Map<String,User>
- instance: UserController
+ loadData(): void
+ resetData(): void
+ login(String username, String password): boolean
+ logout(String username, String password): void
+ createUser(String username, String password): boolean
+ createManagerUser(String username, String password) : boolean
+ isAuthorized(String username, Authorization auth): boolean
+ getUserAuthorizations(String username): Set<Authorization>
+ authorizeUser(String username, Authorization auth): void
0..n
Branch
- LocalTime morningStart
- LocalTime morningEnd
- LocalTime eveningStart
- LocalTime eveningEnd
0..n
Employee
- fullName: String
- ID: String
- bankDetails: String
- hourlySalaryRate: double
- monthlyHours: double
- salaryBonus: double
- employmentDate: LocalDate
- employmentConditions: String
- details: String
- roles: Set<Role>
uses
<<Enum>>
Authorization
HRManager
Cashier
Storekeeper
ShiftManager
LogisticsManager
0..n
holds
User
- username: String
- password: String
- authorizations: List<Authorization>
- loggedIn: boolean
+ isAuthorized(Authorization auth) : boolean
+ login(String password): boolean
+ logout(): void
+ authorize(Authorization auth): void
uses
has
<<Enum>>
Role
Cashier
Storekeeper
ShiftManager
GeneralWorker
SecurityGuard
Steward
Cleaner
Driver
<<Singleton>>
ShiftController
- shifts: Map<String, Map<LocalDate,Map<ShiftType,Shift>>> // branchId to <Date to <ShiftType to Shift>>
- instance: ShiftController
+ getShift(String branchId, LocalDate shiftDate, ShiftType shiftType): Shift
+ shiftExists(String branchId, LocalDate shiftDate, ShiftType shiftType: boolean
+ createShift(String branchId, LocalDate shiftDate, ShiftType shiftType): Shift
+ deleteShift(String branchId, LocalDate shiftDate, ShiftType shiftType): void
+ createWeekShifts(String branchId, LocalDate weekStart): void
+ getWeekShifts(String branchId, LocalDate weekStart): List<Shift[]>
+ setShiftNeededAmount(String branchId, LocalDate shiftDate, ShiftType type, Role role, int amount): void
+ requestShift(Employee employee, String branchId, LocalDate shiftDate, ShiftType shiftType, Role role): void
+ cancelShiftRequest(Employee employee, String branchid, LocalDate shiftDate, ShiftType shiftType, Role role): void
+ setShiftEmployees(String branchId, LocalDate shiftDate, ShiftType shiftType, String role, List<Employee> employees): void
+ getEmployeeShifts(Employee employee): List<Shift[]>
+ approveShift(String branchId, LocalDate shiftDate, ShiftType shiftType): void
+ applyCancelCard(String branchId, LocalDate shiftDate, ShiftType shiftType, String employeeId, String productId): void
+ reportShiftActivity(String branchId, LocalDate shiftDate, ShiftType shiftType, String employeeId, String activity): void
# createBranch(String branchId): void
<<Enum>>
ShiftType
Morning
Evening
has
0..n
Shift
- shiftDate: LocalDate
- shiftType: ShiftType
- isApproved: boolean
- shiftEmployeesAmounts: Map<Role, Integer>
- shiftRequests: Map<Role,List<Employee>>
- shiftWorkers: Map<Role,List<Employee>>
- cancelCardApplies: List<String>
- shiftActivities: List<String>
...


Utils
DateUtils
+ DATE_PATTERN: String
+ DateFormat: DateTimeFormatter
+ getWeekDays(LocalDate d): LocalDate[]
+ getWeekNumber(LocalDate d): int
+ parse(String dateString): LocalDate
+ validDate(String dateInput): boolean
JsonUtils
+ <T> serialize(T object): String
+ <T> deserialize(String json, Type typeOfT): T