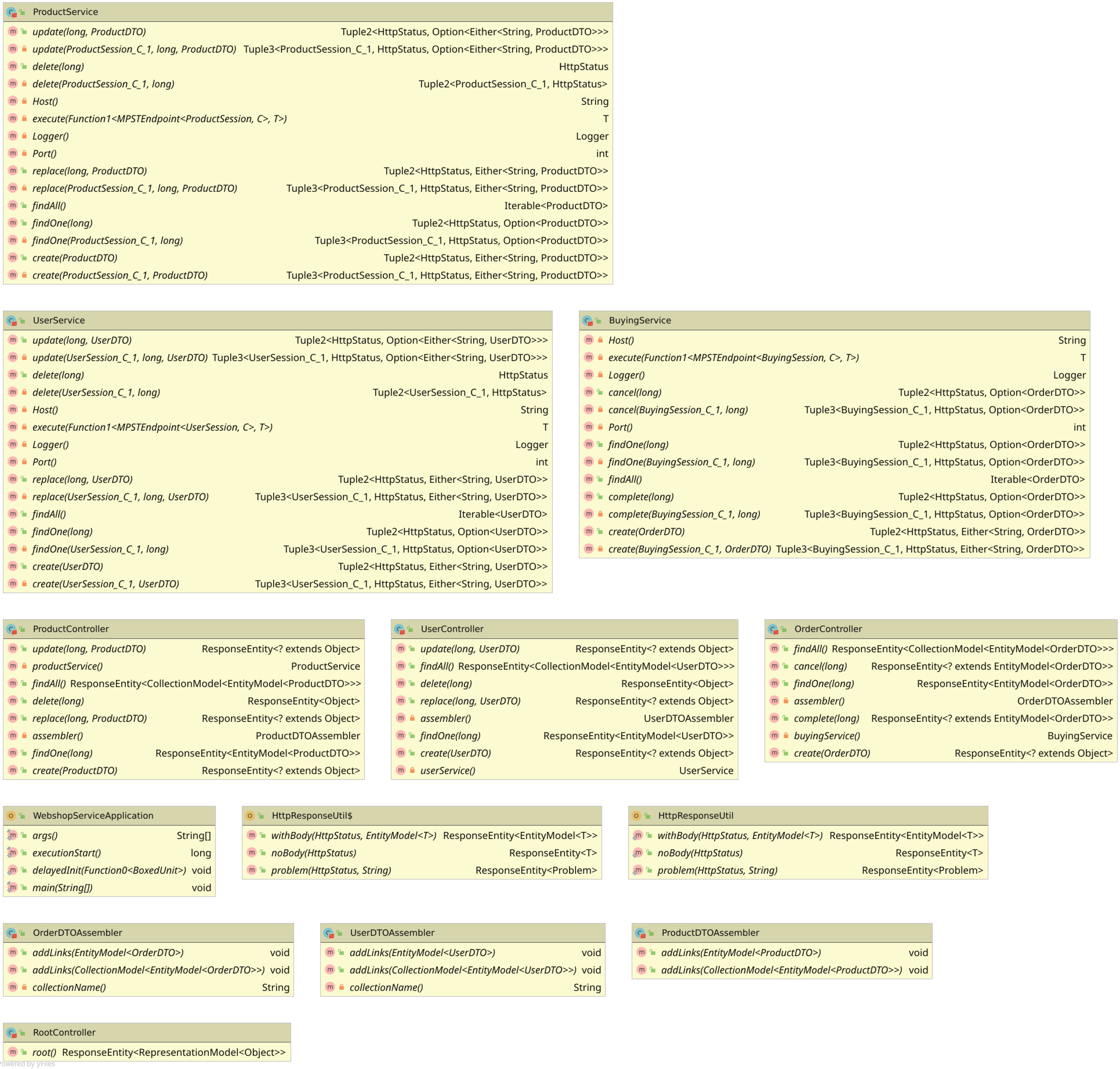

ProductService	
update(long, ProductDTO)	Tuple2<HttpStatus, Option<Either<String, ProductDTO>>>
update(ProductSession_C_1, long, ProductDTO)	Tuple3<ProductSession_C_1, HttpStatus, Option<Either<String, ProductDTO>>>
delete(long)	HttpStatus
delete(ProductSession_C_1, long)	Tuple2<ProductSession_C_1, HttpStatus>
Host()	String
execute(Function1<MPSTEndpoint<ProductSession, C>, T>)	T
Logger()	Logger
Port()	int
replace(long, ProductDTO)	Tuple2<HttpStatus, Either<String, ProductDTO>>
replace(ProductSession_C_1, long, ProductDTO)	Tuple3<ProductSession_C_1, HttpStatus, Either<String, ProductDTO>>
findAll()	Iterable<ProductDTO>
findOne(long)	Tuple2<HttpStatus, Option<ProductDTO>>
findOne(ProductSession_C_1, long)	Tuple3<ProductSession_C_1, HttpStatus, Option<ProductDTO>>
create(ProductDTO)	Tuple2<HttpStatus, Either<String, ProductDTO>>
create(ProductSession_C_1, ProductDTO)	Tuple3<ProductSession_C_1, HttpStatus, Either<String, ProductDTO>>

UserService	
update(long, UserDTO)	Tuple2<HttpStatus, Option<Either<String, UserDTO>>>
update(UserSession_C_1, long, UserDTO)	Tuple3<UserSession_C_1, HttpStatus, Option<Either<String, UserDTO>>>
delete(long)	HttpStatus
delete(UserSession_C_1, long)	Tuple2<UserSession_C_1, HttpStatus>
Host()	String
execute(Function1<MPSTEndpoint<UserSession, C>, T>)	T
Logger()	Logger
Port()	int
replace(long, UserDTO)	Tuple2<HttpStatus, Either<String, UserDTO>>
replace(UserSession_C_1, long, UserDTO)	Tuple3<UserSession_C_1, HttpStatus, Either<String, UserDTO>>
findAll()	Iterable<UserDTO>
findOne(long)	Tuple2<HttpStatus, Option<UserDTO>>
findOne(UserSession_C_1, long)	Tuple3<UserSession_C_1, HttpStatus, Option<UserDTO>>
create(UserDTO)	Tuple2<HttpStatus, Either<String, UserDTO>>
create(UserSession_C_1, UserDTO)	Tuple3<UserSession_C_1, HttpStatus, Either<String, UserDTO>>

BuyingService	
Host()	String
execute(Function1<MPSTEndpoint<BuyingSession, C>, T>)	T
Logger()	Logger
cancel(long)	Tuple2<HttpStatus, Option<OrderDTO>>
cancel(BuyingSession_C_1, long)	Tuple3<BuyingSession_C_1, HttpStatus, Option<OrderDTO>>
Port()	int
findOne(long)	Tuple2<HttpStatus, Option<OrderDTO>>
findOne(BuyingSession_C_1, long)	Tuple3<BuyingSession_C_1, HttpStatus, Option<OrderDTO>>
findAll()	Iterable<OrderDTO>
complete(long)	Tuple2<HttpStatus, Option<OrderDTO>>
complete(BuyingSession_C_1, long)	Tuple3<BuyingSession_C_1, HttpStatus, Option<OrderDTO>>
create(OrderDTO)	Tuple2<HttpStatus, Either<String, OrderDTO>>
create(BuyingSession_C_1, OrderDTO)	Tuple3<BuyingSession_C_1, HttpStatus, Either<String, OrderDTO>>

ProductController	
update(long, ProductDTO)	ResponseEntity<? extends Object>
productService()	ProductService
findAll()	ResponseEntity<CollectionModel<EntityModel<ProductDTO>>>
delete(long)	ResponseEntity<Object>
replace(long, ProductDTO)	ResponseEntity<? extends Object>
assembler()	ProductDTOAssembler
findOne(long)	ResponseEntity<EntityModel<ProductDTO>>
create(ProductDTO)	ResponseEntity<? extends Object>

UserController	
update(long, UserDTO)	ResponseEntity<? extends Object>
findAll()	ResponseEntity<CollectionModel<EntityModel<UserDTO>>>
delete(long)	ResponseEntity<Object>
replace(long, UserDTO)	ResponseEntity<? extends Object>
assembler()	UserDTOAssembler
findOne(long)	ResponseEntity<EntityModel<UserDTO>>
create(UserDTO)	ResponseEntity<? extends Object>
userService()	UserService

OrderController	
findAll()	ResponseEntity<CollectionModel<EntityModel<OrderDTO>>>
cancel(long)	ResponseEntity<? extends EntityModel<OrderDTO>>
findOne(long)	ResponseEntity<EntityModel<OrderDTO>>
assembler()	OrderDTOAssembler
complete(long)	ResponseEntity<? extends EntityModel<OrderDTO>>
buyingService()	BuyingService
create(OrderDTO)	ResponseEntity<? extends Object>

WebshopServiceApplication	
args()	String[]
executionStart()	long
delayedInit(Function0<BoxedUnit>)	void
main(String[])	void

HttpResponseUtil$	
withBody(HttpStatus, EntityModel<T>)	ResponseEntity<EntityModel<T>>
noBody(HttpStatus)	ResponseEntity<T>
problem(HttpStatus, String)	ResponseEntity<Problem>

HttpResponseUtil	
withBody(HttpStatus, EntityModel<T>)	ResponseEntity<EntityModel<T>>
noBody(HttpStatus)	ResponseEntity<T>
problem(HttpStatus, String)	ResponseEntity<Problem>

OrderDTOAssembler	
addLinks(EntityModel<OrderDTO>)	void
addLinks(CollectionModel<EntityModel<OrderDTO>>)	void
collectionName()	String

UserDTOAssembler	
addLinks(EntityModel<UserDTO>)	void
addLinks(CollectionModel<EntityModel<UserDTO>>)	void
collectionName()	String

ProductDTOAssembler	
addLinks(EntityModel<ProductDTO>)	void
addLinks(CollectionModel<EntityModel<ProductDTO>>)	void

RootController	
root()	ResponseEntity<RepresentationModel<Object>>