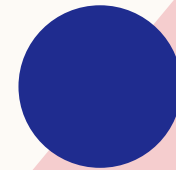


The image features the word "GLAMIFY" in a bold, dark blue, sans-serif font, centered within a white, rounded rectangular shape. This white shape is set against a background composed of three distinct color regions: a light blue area on the left, a light pink area on the right, and a large, dark blue area at the bottom that curves upwards to frame the white shape. The overall design is clean and modern, with a focus on the brand name.

GLAMIFY

AGENDA

- Company Overview
- About Me
- Project Definition
- Area Of Focus
- Project Platforms
- Technologies
- Diagrams
- UI
- Code
- MyWork TimeLine





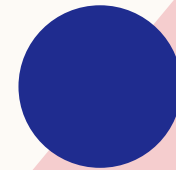
SAUBHAGYAM

COMPANY OVERVIEW

- SAUBHAGYAM Web PVT. LTD Is An innovative Leading Web Development, mobile Application Development, And Services Providing based Company.
- That Has The Specialization And expertise In Web Development For various Types Like Shopify development, Ecommerce Web development, And Many More.
- There Are two Branch INDIA And US

ABOUT ME

Name	Naman Prakashbhai Doshi
Joining_date	04-01-2024
Designation	Python Intern
Mobile_no	9624185617
Email	namandoshi459@gmail.com
Industry_Mentor	Prashant Shastri
Designation	Team leader
Contec	prashant@saubhagyam.org





PROJECT DEFINATIONS

- Glamify Is A State-of-the-art E-commerce Platform Designed To Revolutionize The Beauty And Wellness Shopping Experience.
- As A Comprehensive Web Application, Ios App, And Android App Project.
- Glamify Focuses On Curating A Diverse And High-quality Selection Of Beauty Products.
- Glamify Offers Users A Seamless And Intuitive Interface To Explore And Purchase Skincare, Cosmetic, And Wellness Essentials.



AREAS OF FOCUS

PERSONALIZED BEAUTY SOLUTIONS FOR B2B PARTNERSHIP

- Explore opportunities to collaborate with beauty salons, spas, and wellness centers by offering personalized beauty solutions through Glamify.
- Provide a B2B platform where these businesses can access a curated selection of beauty and skincare products to enhance their service offerings.

PLATFORMS



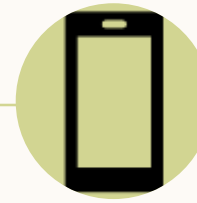
WEB APPLICATION

- We create web applications for broad accessibility through browsers, enabling universal user reach and streamlined updates.



ANDROID APPLICATION

- Android applications provide a dedicated, user-friendly platform, enhancing accessibility and user engagement on mobile devices.



IOS APPLICATION

- IOS applications ensure a tailored user experience on Apple devices, capitalizing on the platform's security and performance, enhancing brand visibility and user engagement.



TECHNOLOGIES

FRONT-END

- REACTJS

BACK-END

- PYTHON
- DJANGO

DATABASE

- POSTGRESQL

HARDWARE

- INTEL CORE I5
- RAM 8GB
- HARD DISK: 1TB

SOFTWARE

- VS CODE
- PGADMIN4
- ANDROID STUDIO
- XCODE
- VISUAL PARADIGM

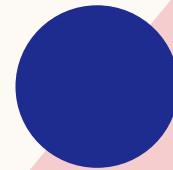
ADMIN

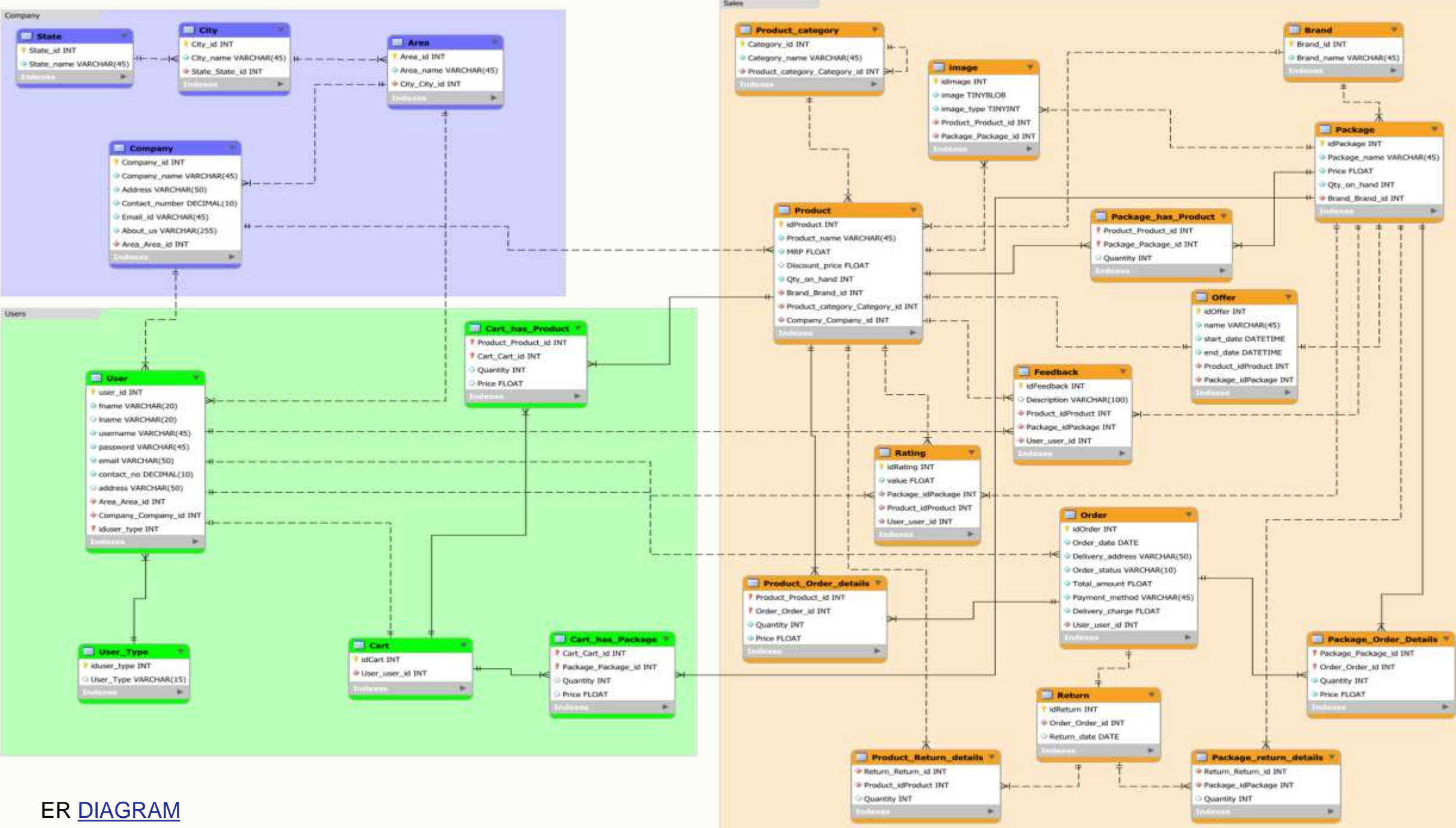
- ❖ Login
- ❖ Manage stock
- ❖ Manage product & categories
- ❖ Manage packages
- ❖ Manage discounts & offers
- ❖ Manage orders
- ❖ Manage payment
- ❖ Manage cancel order
- ❖ Manage sales return
- ❖ Manage feedback & rating

CUSTOMER

- ❖ Registration
- ❖ Login
- ❖ Manage profile
- ❖ Search & View product & Package
- ❖ View Offers & Discount
- ❖ Add to cart
- ❖ Place order
- ❖ Make payment
- ❖ Cancel order
- ❖ Get Invoice
- ❖ View & Give feedback and rating

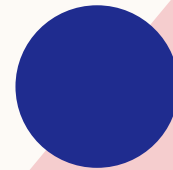
ENTITY RELATIONSHIP

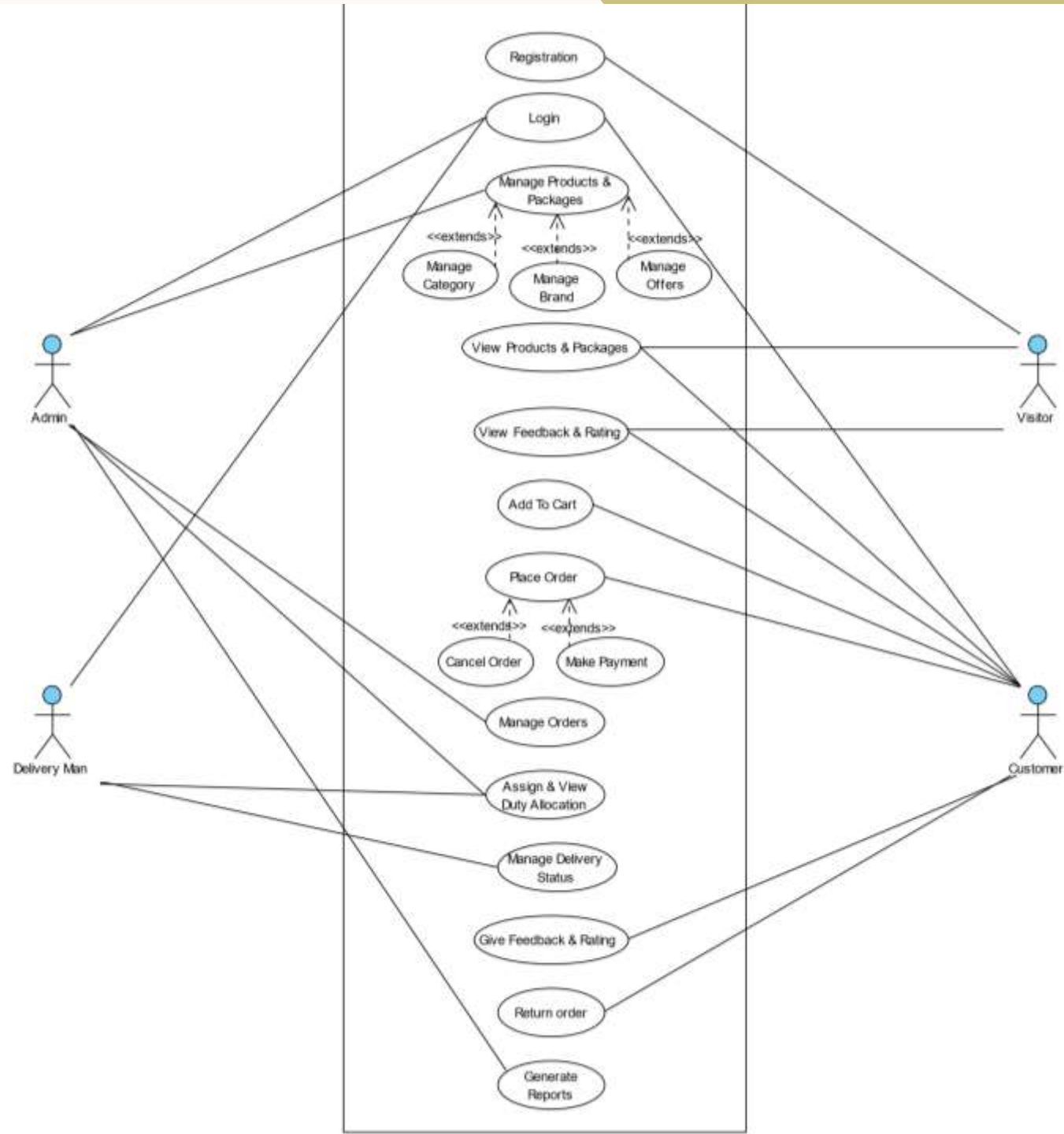




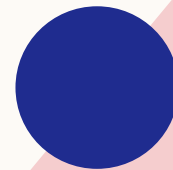
ER DIAGRAM

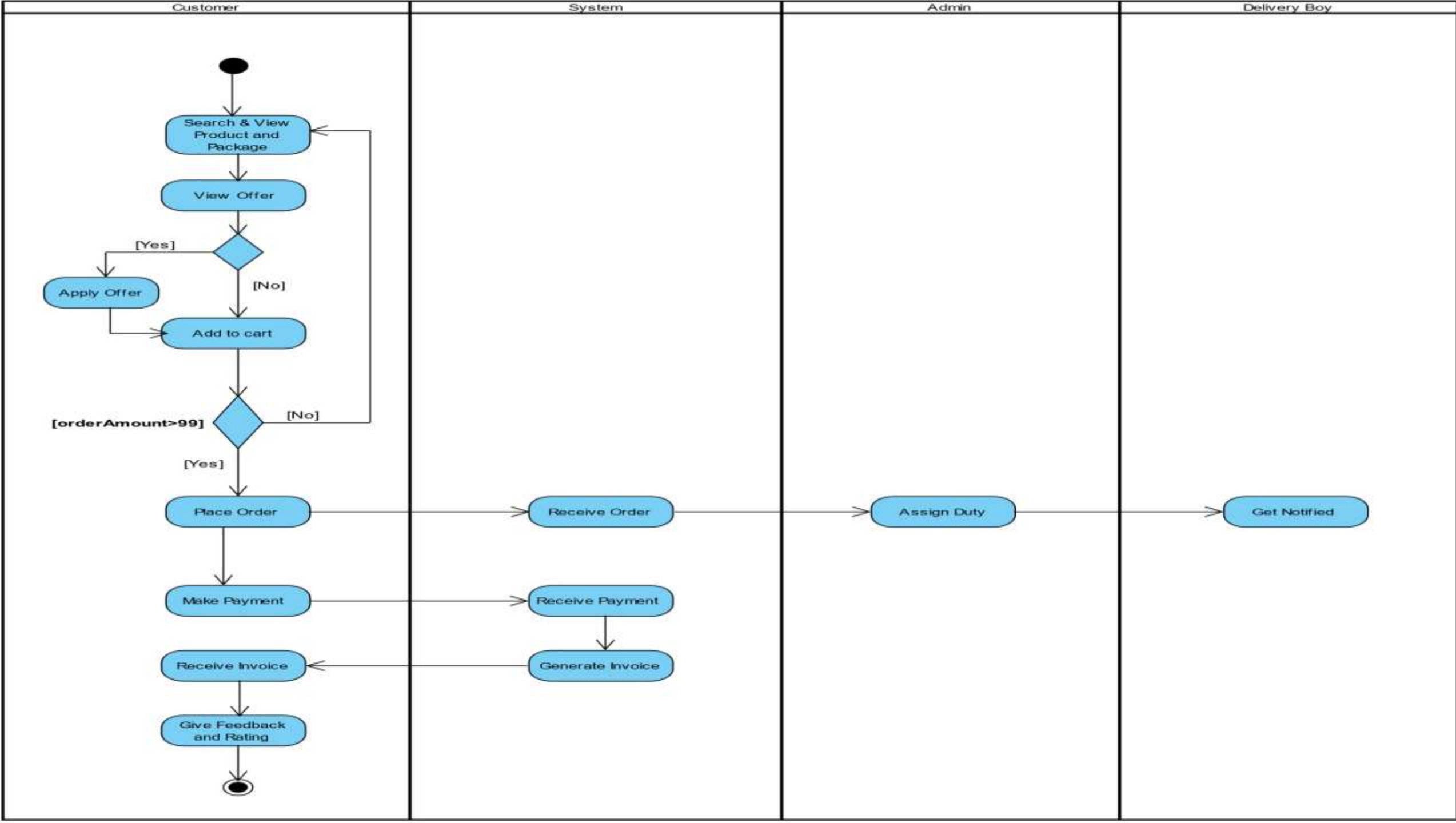
USE CASE



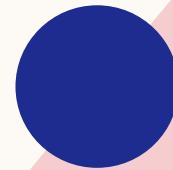


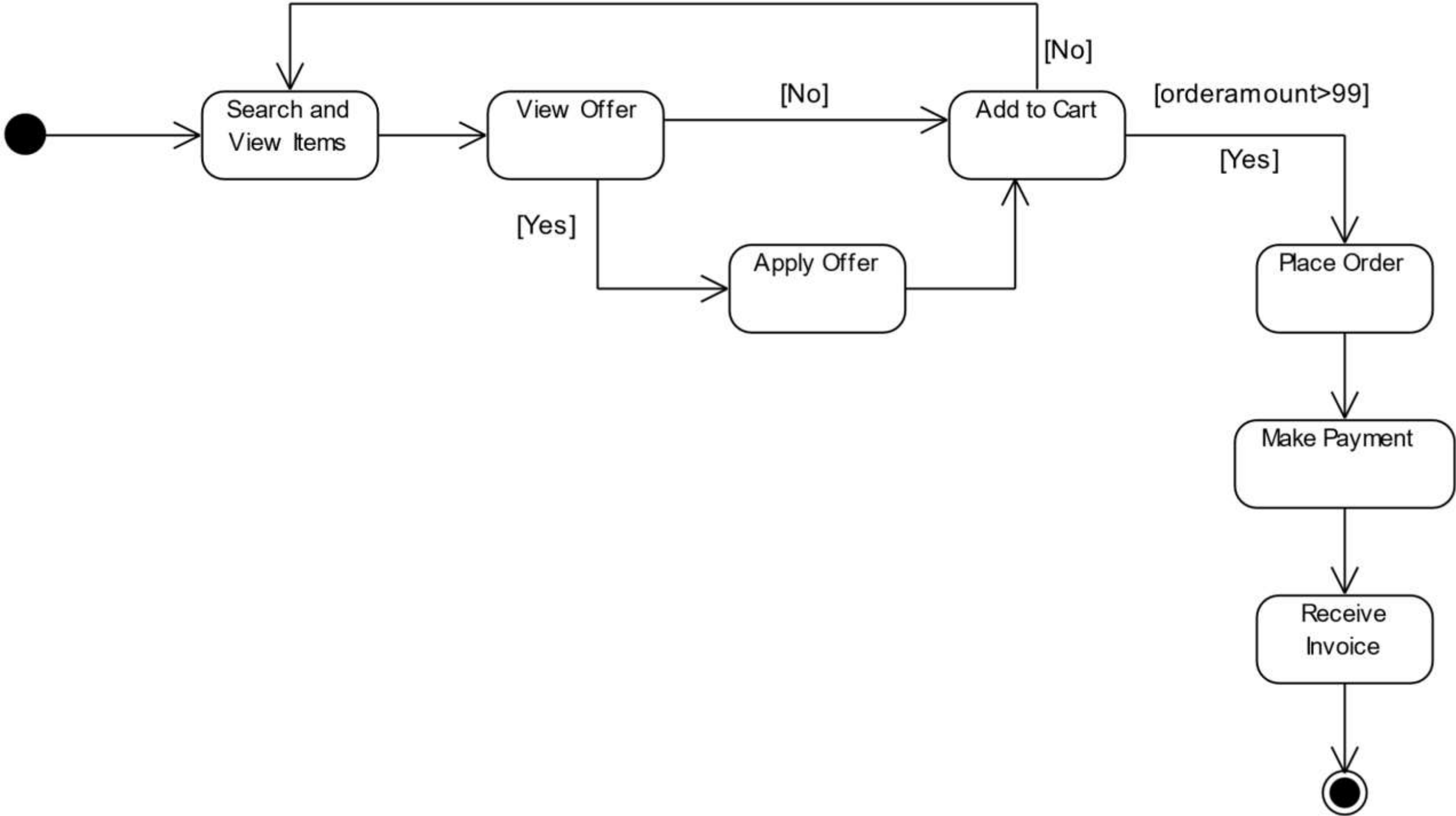
ACTIVITY



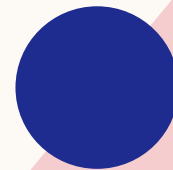


STATE CHART

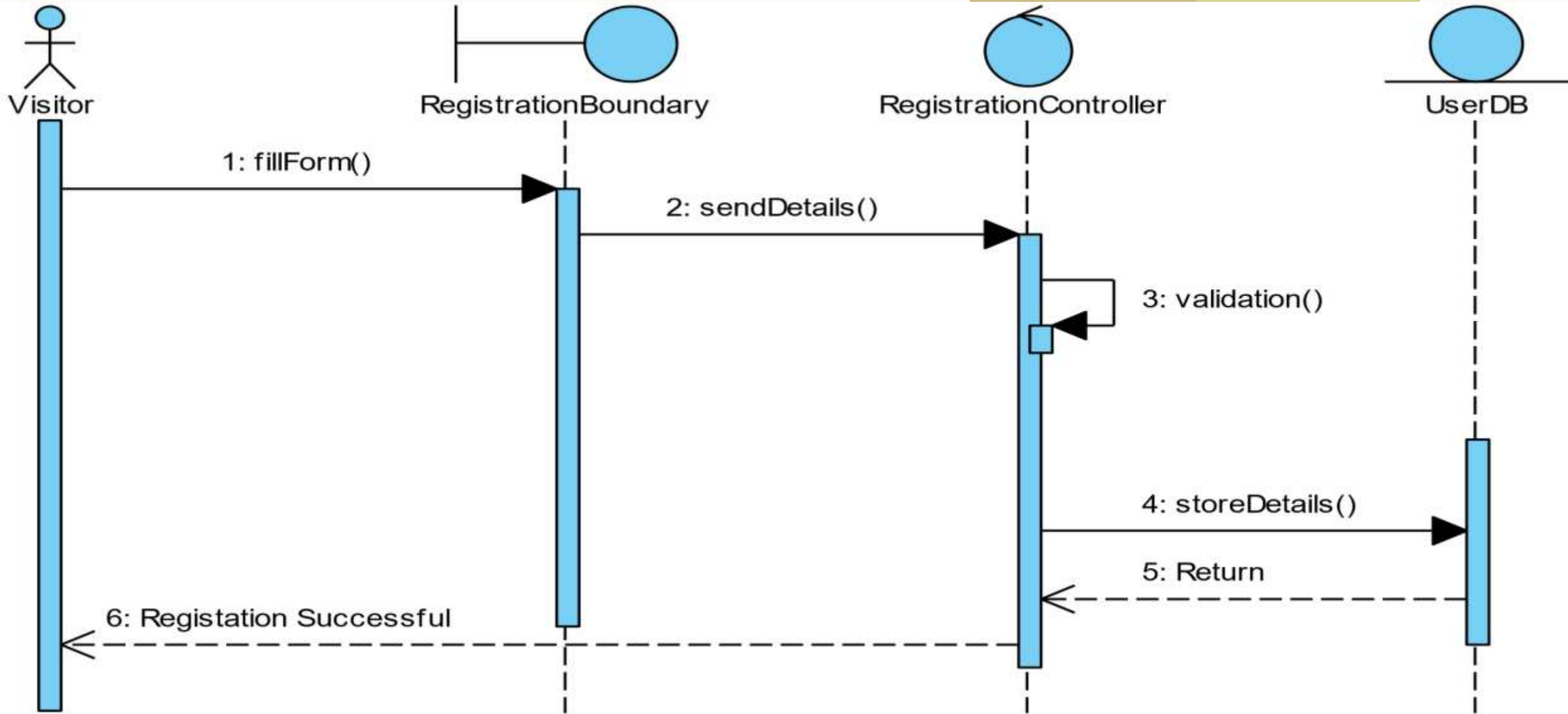




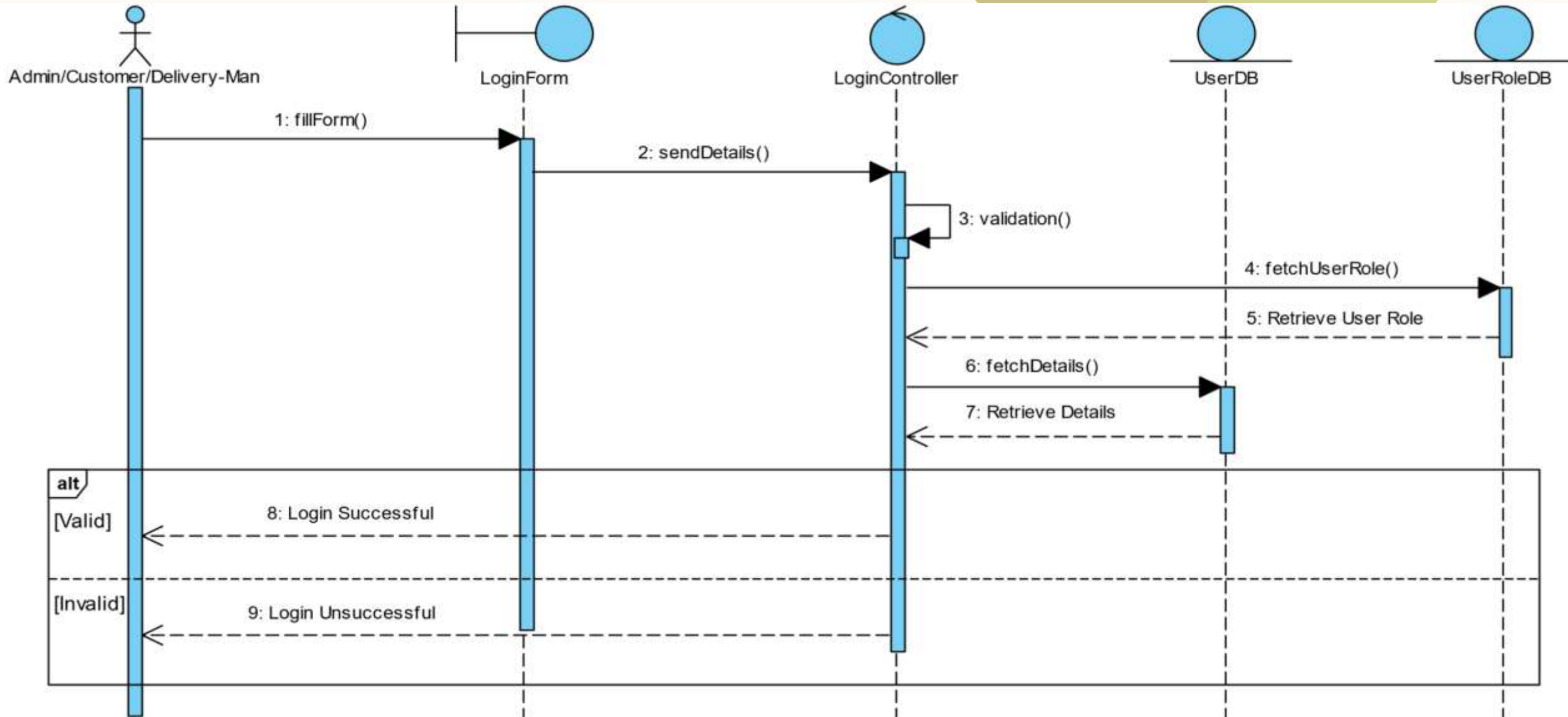
SEQUENCE



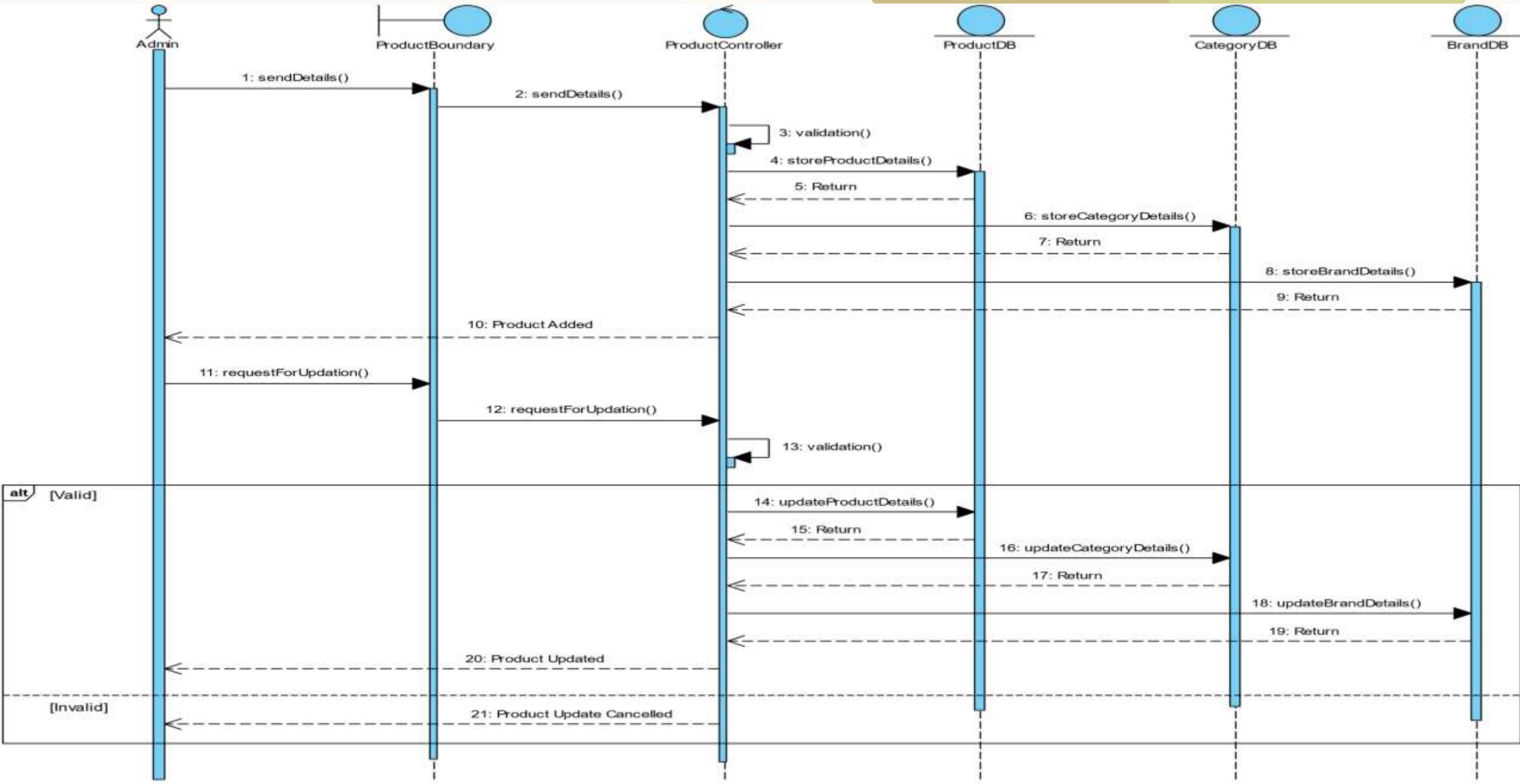
Registration



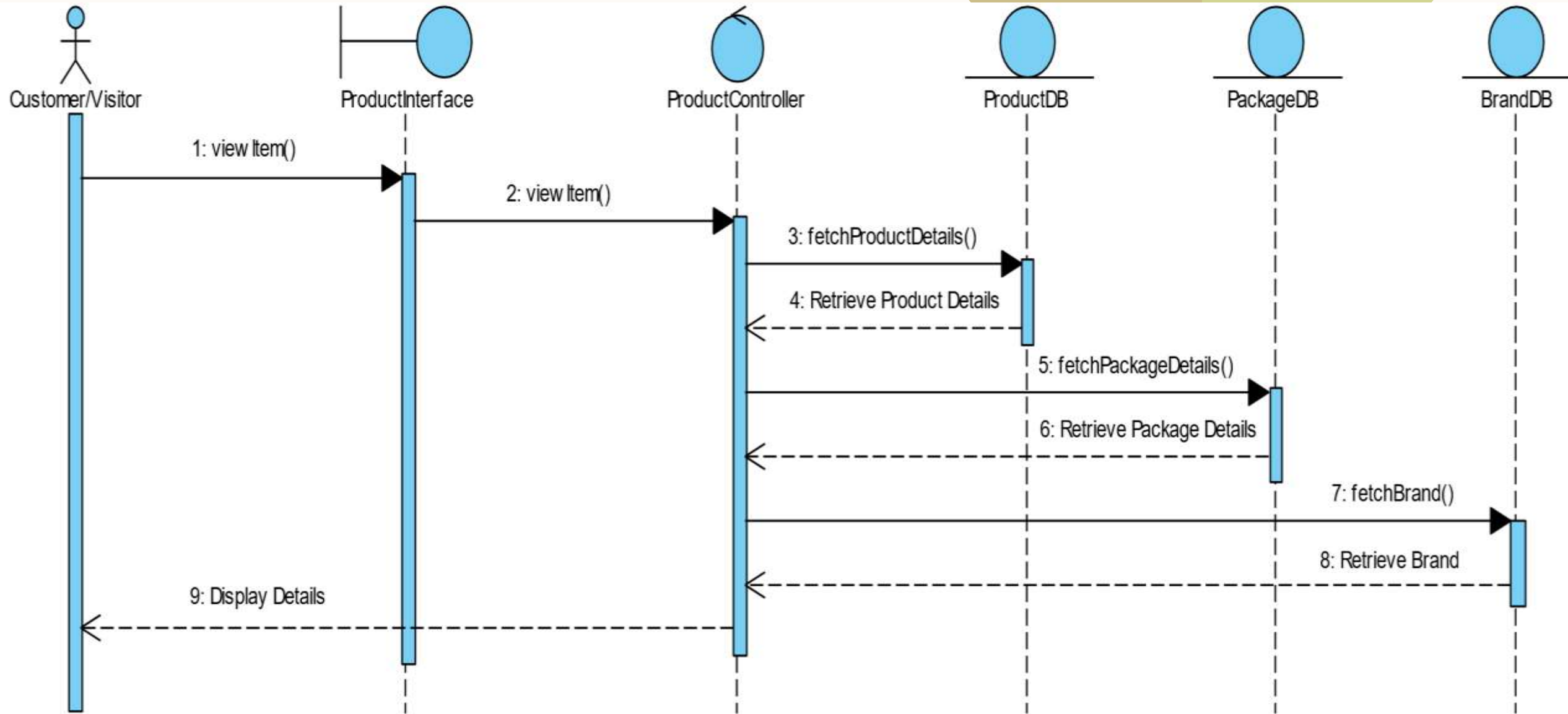
Login



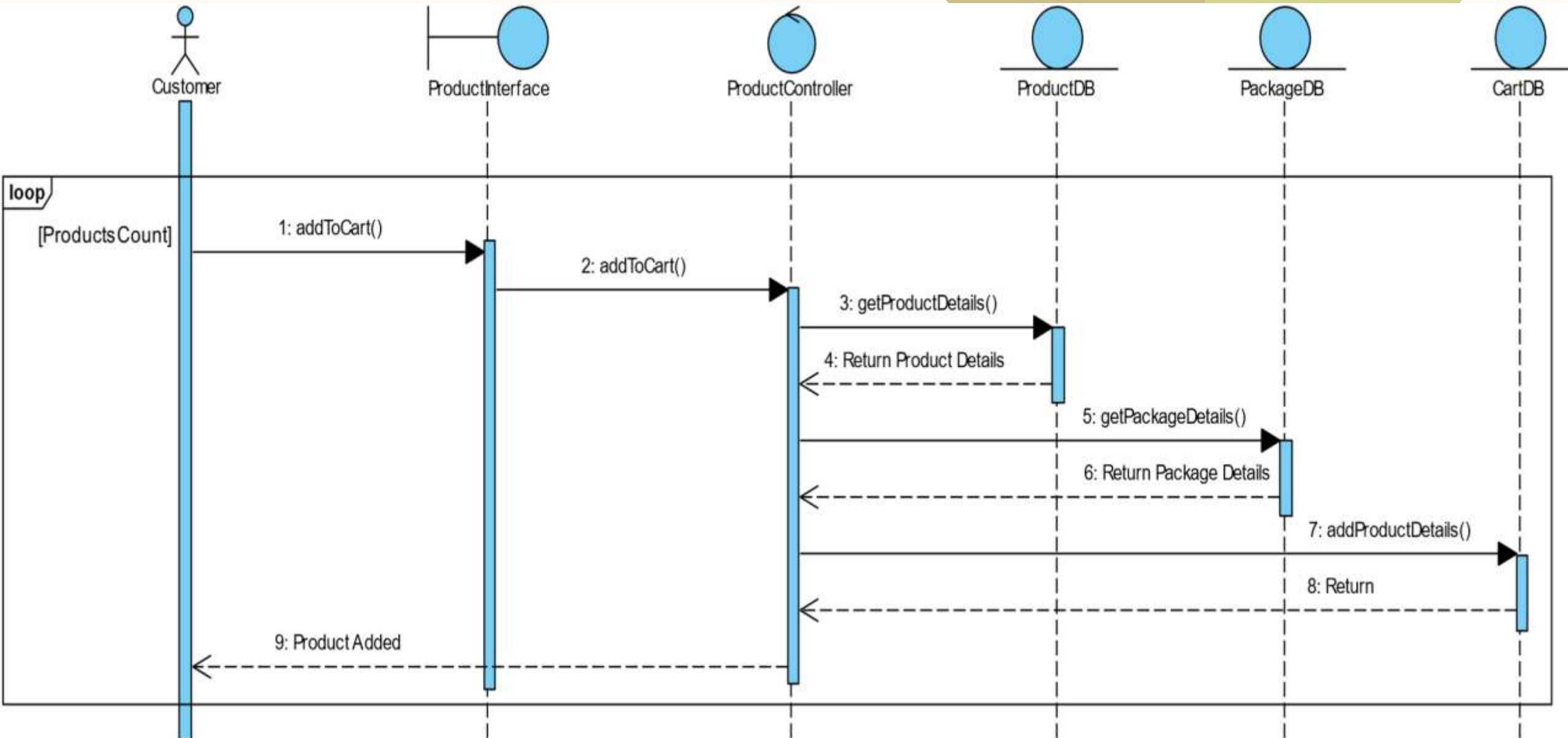
Manage Product



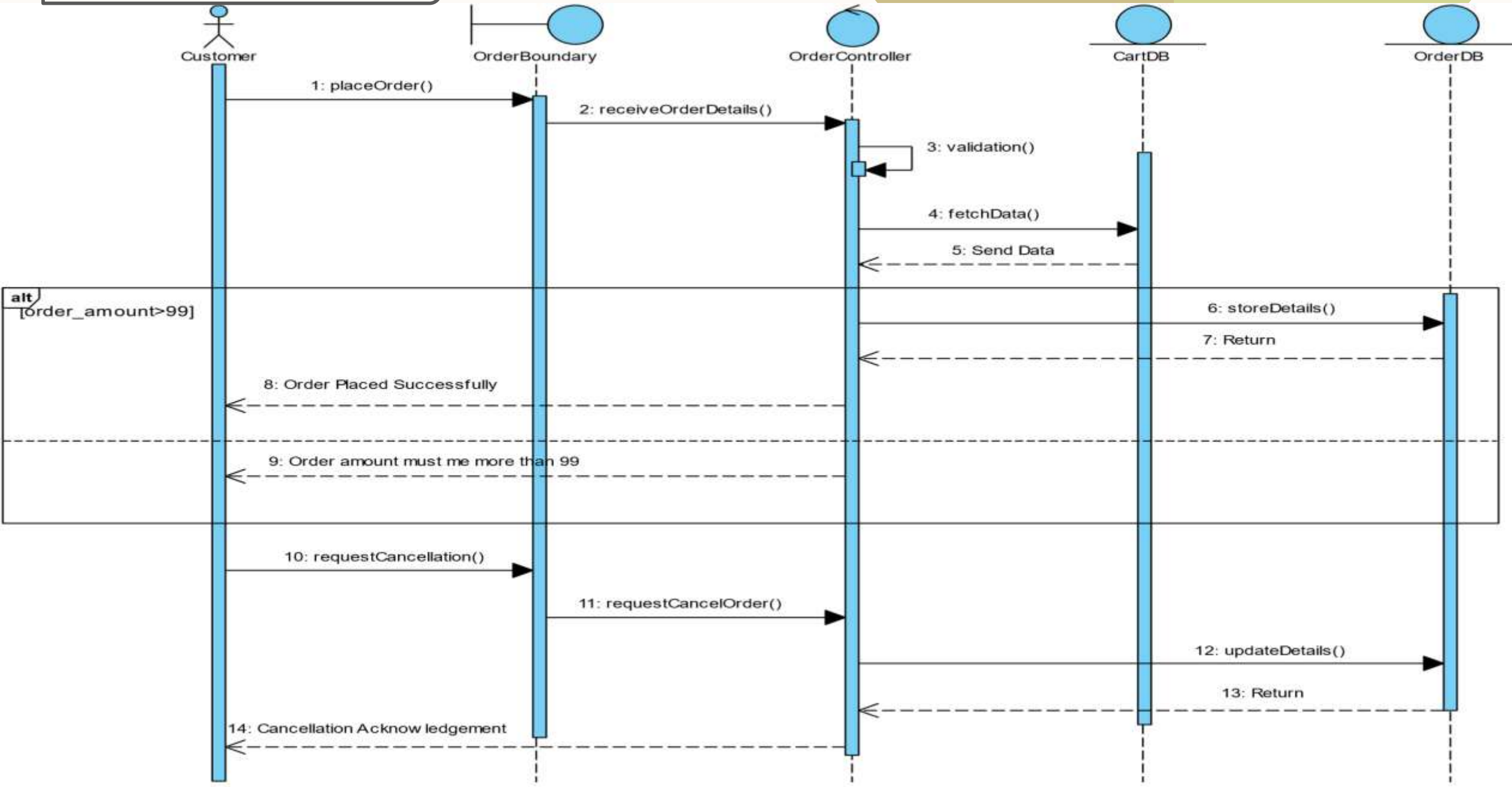
View Items



Add to cart



Place Order



UI



glamify

Create your Account

Name*

testing6

Email*

test6@gmail.com

Password*

•••••



Mobile Number*

1234567891

Sign Up

UI



Sign In to your Account

Email*

test6@gmail.com

Password*

.....



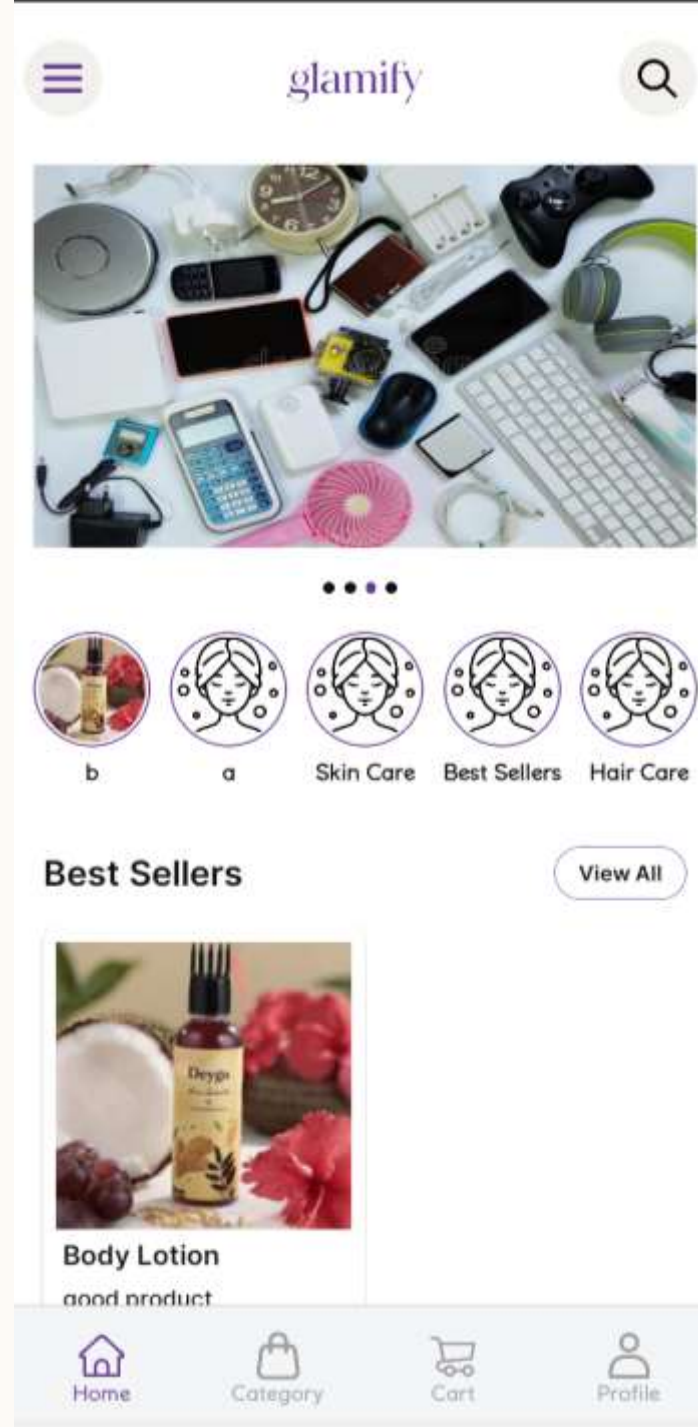
By login, you are agree to Glamify
[Terms & Conditions](#) and [Privacy Policy](#)

Sign In

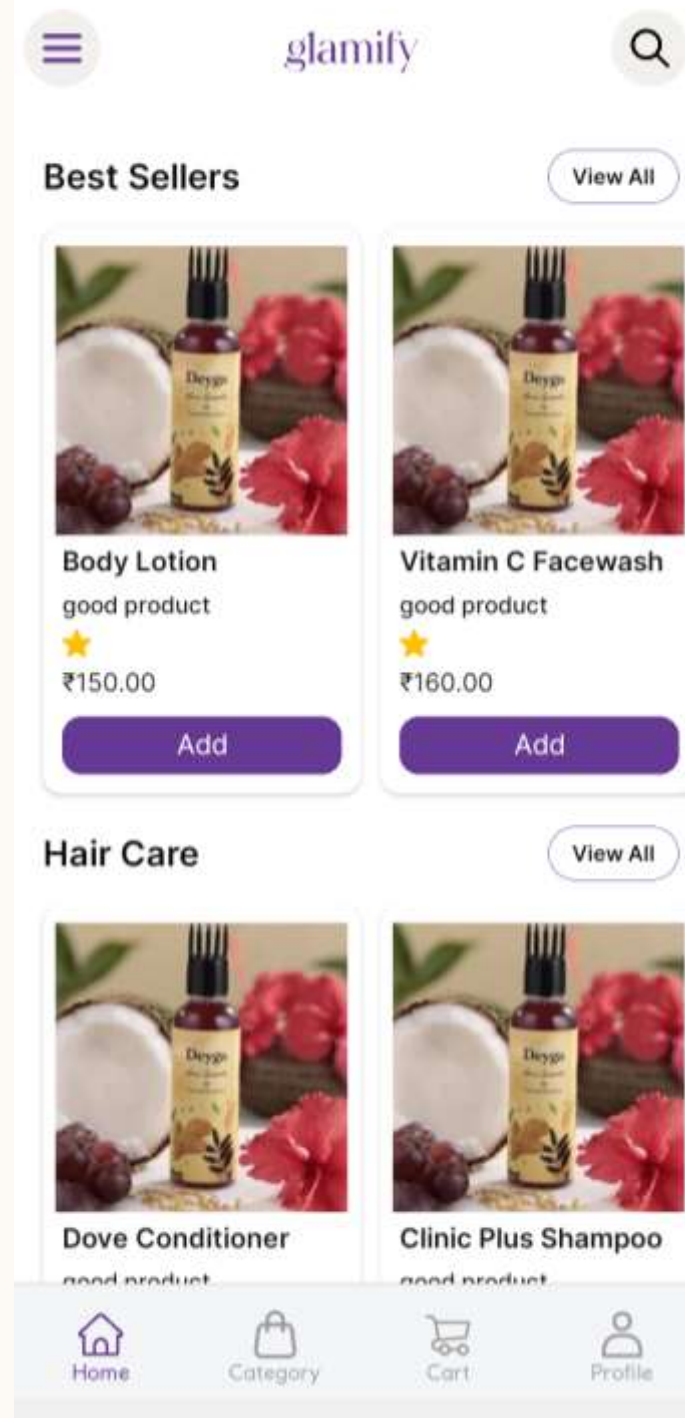
Forgot Password?

Don't have an account? [Sign Up](#)

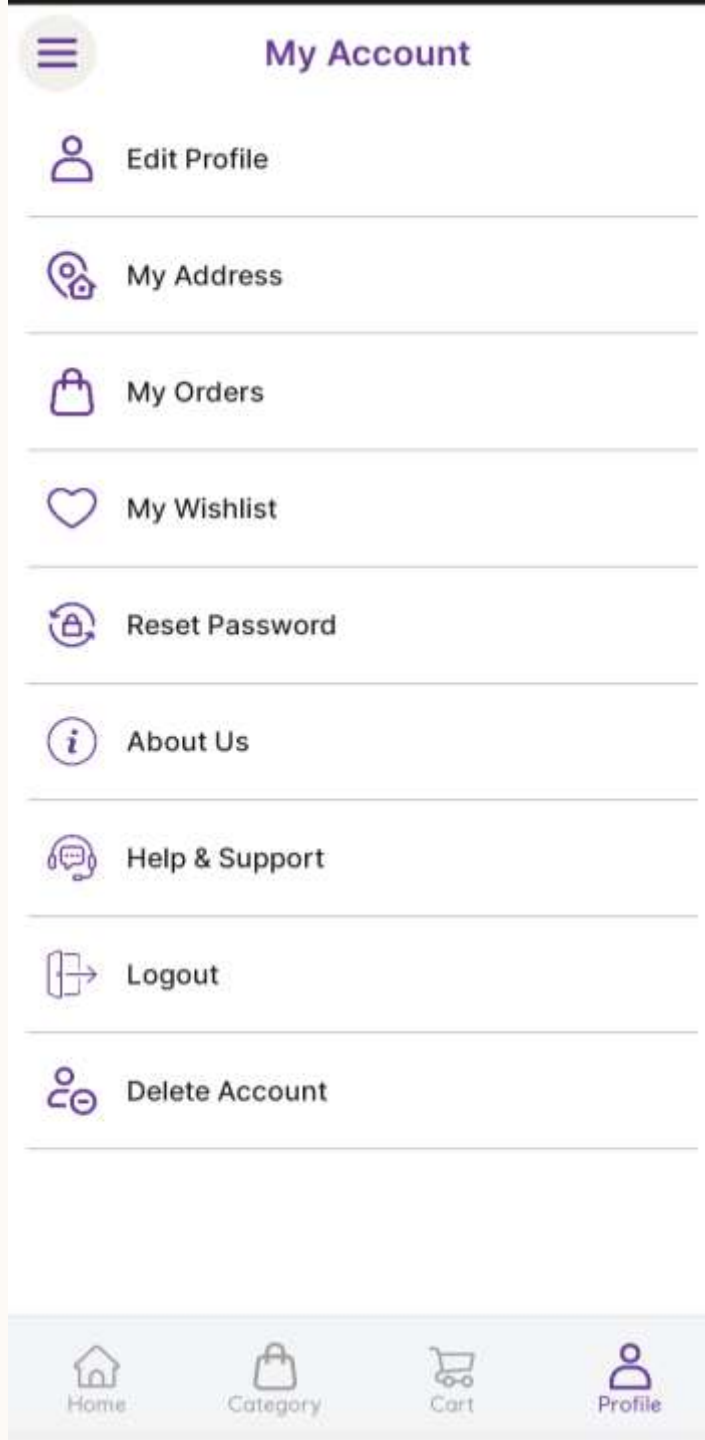
UI



UI



UI



UI



My Wishlist



Hand Wash

Handwash with good quality

₹ 1,500



Hand Wash

Handwash with good quality

₹ 1,500



Hand Wash

Handwash with good quality

₹ 1,500



Hand Wash

Handwash with good quality

₹ 1,500



Hand Wash

Handwash with good quality

₹ 1,500



Requirements Of code

The image shows a Visual Studio Code editor window with the following components:

- Top Bar:** Contains the menu bar (File, Edit, Selection, View, Go, Run, ...) and a search bar with the text "ecommerce-be-main".
- Left Panel (EXPLORER):** Displays the file tree for a project named "ECOMMERCE...". The tree structure is as follows:
 - ECOMMERCE...
 - .vscode
 - requirements
 - base.txt** (selected)
 - local.txt
 - prod.txt
 - server
 - authentication
 - migrations
 - __init__.py
 - admin.py
 - apps.py
 - models.py
 - serializers.py
 - tests.py
 - urls.py
 - views.py
 - constants
 - master
 - media
 - middleware
 - orders
 - server
 - utils
 - manage.py
 - .gitignore

- Editor Area:** Shows the content of the selected file "base.txt" under the "requirements" directory. The file contains the following Django requirements:

```
1 Django~=4.2
2 psychopg[binary]==3.1.15
3
4 djangorestframework==3.14.0
5 django-cors-headers==4.3.1
6 django-filter==23.5
7 djangorestframework-simplejwt==5.3.1
8
9 pillow==10.1.0
10 django-imagekit==5.0.0
11
12 drf-yasg
13 django-debug-toolbar==4.2.0
14
```
- Bottom Panel:** Displays the status bar with the following information: "Ln 8, Col 1", "Spaces: 4", "UTF-8", "LF", "pip requirements", and a Python extension loading status.

EXPLORER

ECOMMERCE...

.vscode

requirements

base.txt

local.txt

prod.txt

server

authentication

constants

master

media

middleware

orders

server

__pycache__

settings

__pycache__

base.py

dev.py

prod.py

__init__.py

asgi.py

urls.py

wsgi.py

utils

manage.py

__main__.py

OUTLINE

TIMELINE

views.py

base.py

Welcome to GitLens

server > server > settings > base.py > ...

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

"django.contrib.messages.context_processors.messages",

],

},

],

WSGI_APPLICATION = "server.wsgi.application"

ASGI_APPLICATION = "server.asgi.application"

DATABASES = {

"default": {

"ENGINE": "django.db.backends.postgresql",

"USER": "postgres",

"NAME": "glamify-db",

"PASSWORD": "1234",

"HOST": "localhost",

"PORT": "5432",

}

}

Internationalization

<https://docs.djangoproject.com/en/4.2/topics/i18n/>

LANGUAGE_CODE = "en-us"

TIME_ZONE = "UTC"

USE_I18N = True

USE_TZ = True

Static files (CSS, JavaScript, Images)

Ln 1, Col 1

Spaces: 4

UTF-8

LF

Python

3.12.1 64-bit

Go Live

Blackbox

Main URLs

File Edit Selection View Go Run

server > server > urls.py > ...

EXPLORER

ECOMMERCE-BE-MAIN

.vscode

requirements

base.txt

local.txt

prod.txt

server

authentication

constants

master

media

middleware

orders

server

__pycache__

settings

__pycache__

base.py

dev.py

prod.py

__init__.py

asgi.py

urls.py

wsgi.py

utils

manage.py

__main__.py

OUTLINE

TIMELINE

views.py

urls.py

Welcome to GitLens

```
38
39 urlpatterns = [
40     path("admin/", admin.site.urls),
41     path(name="auth", route="api/auth/", view=include("authentication.urls")),
42     path(name="master", route="api/", view=include("master.urls")),
43     path(name="order", route="api/", view=include("master.urls")),
44     path(
45         "doc/",
46         schema_view.with_ui("swagger", cache_timeout=0),
47         name="schema-swagger-ui",
48     ),
49     path("redoc/", schema_view.with_ui("redoc", cache_timeout=0), name="schema-redoc"),
50     path("json/", schema_view.without_ui(cache_timeout=0), name="schema-json"),
51 ] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
52
53 admin.site.AdminSite.site_title = "Ecommerce Admin"
54 admin.site.AdminSite.site_header = "Ecommerce Dashboard"
55 admin.site.AdminSite.index_title = "Ecommerce Admin Panel"
56
```

Ln 31, Col 65 Spaces: 4 UTF-8 LF Python 3.12.1 64-bit Go Live Blackbox

Authentication App Model

File Edit Selection View Go Run ... ecommerce-be-main

EXPLORER

- ECOMMERCE-BE-MAIN
 - .vscode
 - requirements
 - base.txt
 - local.txt
 - prod.txt
 - server
 - authentication
 - migrations
 - __init__.py
 - admin.py
 - apps.py
 - models.py**
 - serializers.py
 - tests.py
 - urls.py
 - views.py
 - constants
 - master
 - media
 - banners
 - category
 - icons\categories
 - products
 - middleware
 - orders
 - server

models.py

server > authentication > models.py > ...

Click here to ask Blackbox to help you code faster

```
1 from django.contrib.auth.models import (AbstractBaseUser, BaseUserManager,
2                                     Group, PermissionsMixin)
3 from django.db import models
4
5
6 class AccountManager(BaseUserManager):
7     use_in_migrations = True
8
9     def _create_user(self, email, password=None, **extra_fields):
10         if not email:
11             raise ValueError("The Email field must be set")
12
13         email = self.normalize_email(email)
14         user = self.model(email=email, **extra_fields)
15         user.set_password(password)
16         user.save(using=self._db)
17         return user
18
19     def create_user(self, email, password=None, **extra_fields):
20         extra_fields.setdefault("is_active", True)
21         extra_fields.setdefault("is_staff", False)
22         extra_fields.setdefault("is_superuser", False)
23         return self._create_user(email, password, **extra_fields)
24
25     def create_staff_user(self, email, password=None, **extra_fields):
26         extra_fields.setdefault("is_staff", True)
27         extra_fields.setdefault("is_superuser", False)
28
29         if extra_fields.get("is_staff") is not True:
30             raise ValueError("Staff user must have is_staff=True.")
31
```

Ln 1 Col 1 Spaces: 4 UTF-8 LF Python 3.12.1 64-bit Go Live Blackbox

Authentication API

File Edit Selection View Go Run ... ecommerce-be-main

EXPLORER

- ECOMMERCE-BE-MAIN
 - .vscode
 - requirements
 - base.txt
 - local.txt
 - prod.txt
 - server
 - authentication
 - migrations
 - serializers.py
 - tests.py
 - urls.py
 - views.py
 - constants
 - master
 - media
 - banners
 - category
 - icons
 - categories
 - products
 - middleware
 - orders
 - server

server > authentication > serializers.py > ...

```
1 from rest_framework import serializers
2
3 from .models import AccountModel
4
5 # from django.contrib.auth.password_validation import validate_password
6
7
8 class AccountSerializer(serializers.ModelSerializer):
9     password = serializers.CharField(
10         write_only=True,
11         required=True,
12         # validators=[validate_password]
13     )
14
15     def create(self, validated_data):
16         user = AccountModel.objects.create(
17             email=validated_data["email"],
18             name=validated_data["name"],
19             mobile=validated_data["mobile"],
20         )
21
22         user.set_password(validated_data["password"])
23         user.save()
24         return user
25
26 class Meta:
27     model = AccountModel
28     fields = ["name", "email", "mobile", "password"]
29     extra_kwargs = {"email": {"error_messages": {"required": "Email is required!"}}}
```

Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.12.1 64-bit Go Live Blackbox

Authentication App Views

File Edit Selection View Go Run

ecommerce-be-main

EXPLOSER

ECOMMERCE-BE-MAIN

server > authentication > views.py > LoginAPI > post

views.py 1 X

server > authentication > views.py > LoginAPI > post

18 class UserRegistrationAPI(APIView):

19 permission_classes = (AllowAny,)

20 serializer_class = AccountSerializer

21

22 @swagger_auto_schema(

23 tags=["Authentication"],

24 request_body=AccountSerializer,

25 operation_description="Endpoint for user registration.",

26)

27 def post(self, request):

28 serializer = self.serializer_class(data=request.data)

29 if serializer.is_valid():

30 serializer.save()

31 return Response(

32 data={

33 "status": True,

34 "message": "Your account has been registered successfully!",

35 },

36 status=STATUS_CODES.CREATED,

37)

38 else:

39 return Response(

40 data={"status": False, "message": serializer.errors},

41 status=STATUS_CODES.BAD_REQ,

42)

43

44

45 class LoginAPI(APIView):

46 @swagger_auto_schema(

47 tags=["Authentication"],

48 request_body=openapi.Schema(

49 type=openapi.TYPE_OBJECT,

Ln 48, Col 37 Spaces: 4 UTF-8 LF Python 3.12.1 64-bit Go Live Blackbox

Authentication App URLs

The image shows a Visual Studio Code editor window with the file explorer on the left and the code editor in the center. The file explorer shows the project structure for 'ecommerce-be-main', with the 'authentication' app selected. The 'urls.py' file is highlighted in the file explorer and is also the active file in the code editor. The code in 'urls.py' defines the URL patterns for the authentication app, including routes for user registration, login, password update, visitor validation, and visitor ID generation.

server > authentication > urls.py > ...

```
1 from django.urls import path
2
3 from .views import (
4     GenerateVisitorAPI,
5     LoginAPI,
6     UpdatePasswordAPI,
7     UserRegistrationAPI,
8     ValidateVisitorAPI,
9 )
10
11 urlpatterns = [
12     path(route="register/", view=UserRegistrationAPI.as_view(), name="user-register"),
13     path(route="login/", view=LoginAPI.as_view(), name="login"),
14     path(
15         route="update-password/",
16         view=UpdatePasswordAPI.as_view(),
17         name="update-password",
18     ),
19     path(
20         route="validate-visitor-id/",
21         view=ValidateVisitorAPI.as_view(),
22         name="validate-visitor-id",
23     ),
24     path(
25         route="generate-visitor-id/",
26         view=GenerateVisitorAPI.as_view(),
27         name="generate-visitor-id",
28     ),
29 ]
30
```

Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.12.1 64-bit Go Live Blackbox

Models for Order App

File Edit Selection View Go Run ... ecommerce-be-main

EXPLORER

- ECOMMERCE...
- .vscode
- requirements
 - base.txt
 - local.txt
 - prod.txt
- server
 - authentication
 - constants
 - master
 - media
 - middleware
 - orders
 - migrations
 - serializers
 - CartSerializer.py
 - views
 - __init__.py
 - admin.py
 - apps.py
 - models.py**
 - views.py
 - server
 - utils
 - manage.py
 - .gitignore

models.py

```
server > orders > models.py > ...  
Click here to ask Blackbox to help you code faster  
1 from authentication.models import AccountModel, VisitorModel  
2 from django.db import models  
3 from master.models import ProductModel, TimestampModel  
4  
5 # Create your models here.  
6  
7  
8 class GstTax(TimestampModel):  
9     sgst = models.FloatField(default=0.0)  
10    cgst = models.FloatField(default=0.0)  
11  
12  
13 class CartItem(TimestampModel):  
14     product_name = models.ForeignKey(ProductModel, on_delete=models.CASCADE)  
15     product_quantity = models.PositiveIntegerField(default=1)  
16  
17  
18 class Cart(TimestampModel):  
19     items = models.ManyToManyField(CartItem)  
20     user = models.ForeignKey(  
21         AccountModel, on_delete=models.CASCADE, blank=True, null=True)  
22     visitor = models.ForeignKey(  
23         VisitorModel, on_delete=models.CASCADE, blank=True, null=True)  
24  
25  
26     # @property  
27     # def total_qty(self):  
28     #     return sum(item.adult_quantity + item.child_quantity for item in self.items.all())  
29  
30  
31     # @property
```

Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.12.1 64-bit Go Live Blackbox

Add All Media

VS

File Edit Selection View Go Run ...

← →

ecommerce-be-main

□ □ □ 0%

—

□

×

EXPLORER

▼ ECOMMERCE-BE-MAIN

▼ server

▼ media

▼ banners

image_2024_01_02T13_25_46_399Z.png

product.png

product2.png

special_offer.png

test_img.jpg

▼ category

images_25zLuWa.jpg

images_89pHE2G.jpg

images_96rCih9.jpg

images_ekVFHaG.jpg

images_kGbIQ4R.jpg

images_lawH5qk.jpg

images_qoHilDF.jpg

images_VqNmeHi.jpg

images_W4Ra2Eu.jpg

images_Yc28epB.jpg

images.jpg

▼ icons \ categories

0d721e88f5cb4e39bf8a847339097683.png

0d7364dc6ec9400985ab2b3f9cb1deb4.jpg

8f3045a167b941abbaa6f001e8eeeb42.jpg

9fe7354e3fdb4deab03202af2e8e7a11.jpg

> OUTLINE

> TIMELINE

...

+

+

↺

□

Show All Commands

Ctrl + Shift + P

Go to File

Ctrl + P

Find in Files

Ctrl + Shift + F

Toggle Full Screen

F11

Show Settings

Ctrl + ,

0 0 0

Share Code Link

Explain Code

Comment Code

Find Bugs

Code Chat

Search Error

Go Live

Blackbox

VS

API in master App

The image shows a Visual Studio Code editor window with a dark theme. The Explorer sidebar on the left displays the project structure for 'ecommerce-be-main'. The file 'CategorySerializer.py' is selected and highlighted. The main editor area shows the code for this file, which includes imports from 'rest_framework' and 'master.models', and defines three serializer classes: 'SubCategoryReadSerializer', 'SubCategoryWriteSerializer', and 'CategoryReadSerializer'. The status bar at the bottom indicates the file is at line 1, column 1, with 4 spaces, in UTF-8 encoding, using Python 3.12.1 64-bit. The Blackbox logo is visible in the bottom right corner.

File Edit Selection View Go Run ...

ecommerce-be-main

EXPLORER

ECOMMERCE-BE-MAIN

- requirements
 - local.txt
 - prod.txt
- server
 - authentication
 - constants
 - master
 - filters
 - ProductFilter.py
 - migrations
 - serializers
 - CategorySerializer.py**
 - HomePageSerializer.py
 - ProductSerializer.py
 - views
 - __init__.py
 - admin.py
 - apps.py
 - models.py
 - tests.py
 - urls.py
 - media
 - banners
 - category
 - icons
 - categories
 - products
- OUTLINE
- TIMELINE

CategorySerializer.py

server > master > serializers > CategorySerializer.py > ...

Click here to ask Blackbox to help you code faster

```
1 from rest_framework import serializers
2
3 from master.models import CategoryModel, SubCategoryModel
4
5
6 class SubCategoryReadSerializer(serializers.ModelSerializer):
7     class Meta:
8         model = SubCategoryModel
9         exclude = ["original_icon"]
10
11
12 class SubCategoryWriteSerializer(serializers.ModelSerializer):
13     class Meta:
14         model = SubCategoryModel
15         fields = "__all__"
16
17
18 class CategoryReadSerializer(serializers.ModelSerializer):
19     def to_representation(self, instance):
20         request = self.context.get("request")
21         representation = super().to_representation(instance)
22         subcategories = SubCategoryModel.objects.filter(category=instance.id)
23         if subcategories:
24             representation["sub_category"] = SubCategoryReadSerializer(
25                 subcategories, many=True, context={"request": request}
26             ).data
27         else:
28             representation["sub_category"] = []
29         return representation
30
31 class Meta:
```

Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.12.1 64-bit Go Live Blackbox

Master App Urls

File

Edit

Selection

View

Go

Run

...

←

→

ecommerce-be-main

□

□

□

□

—

□

×

EXPLORER

...

▼ ECOMMERCE-BE-MAIN

▼ requirements

local.txt

prod.txt

▼ server

authentication

constants

▼ master

▼ filters

ProductFilter.py

migrations

serializers

views

__init__.py

admin.py

apps.py

models.py

tests.py

urls.py

▼ media

banners

category

icons \categories

products

middleware

orders

server

▼ OUTLINE

▼ TIMELINE

urls.py

×

server > master > urls.py > ...

23 from master.views.HomePageView import HomeBannerListAPIView, HomeCategoryListAPIView

24

25 urlpatterns = [

26 path("categories/", CategoryListAPIView.as_view(), name="category-list"),

27 path(

28 "categories/<int:pk>/",

29 CategoryRetrieveAPIView.as_view(),

30 name="category-detail",

31),

32 path("subcategories/", SubCategoryListAPIView.as_view(), name="subcategory-list"),

33 path(

34 "subcategories/<int:pk>/",

35 SubCategoryRetrieveAPIView.as_view(),

36 name="subcategory-detail",

37),

38 path("products/", SiteProductListAPIView.as_view(), name="product-site-list"),

39 path(

40 "products/<int:pk>/",

41 SiteProductRetrieveAPIView.as_view(),

42 name="product-site-detail",

43),

44 # ----- ADMIN ACCESSIBLE URLS -----

45 path("categories/create/", CategoryCreateAPIView.as_view(), name="category-create"),

46 path(

47 "categories/<int:pk>/update/",

48 CategoryUpdateAPIView.as_view(),

49 name="category-update",

50),

51 path(

52 "subcategories/create/",

53 SubCategoryCreateAPIView.as_view(),

Ln 1, Col 1

Spaces: 4

UTF-8

LF

Python

3.12.1 64-bit

Go Live

Blackbox

Constants Files Code

File

Edit

Selection

View

Go

Run

...

←

→

ecommerce-be-main

□

□

□

□

—

□

×

EXPLORER

...

▼ ECOMMERCE-BE-MAIN

▼ .vscode

▼ requirements

base.txt

local.txt

prod.txt

▼ server

▼ authentication

▼ constants

exceptions.py

status.py

▼ master

▼ media

▼ banners

▼ category

▼ icons\categories

▼ products

▼ middleware

▼ orders

▼ server

▼ utils

manage.py

.gitignore

▼ OUTLINE

▼ TIMELINE

status.py

×

server > constants > status.py > ...

Click here to ask Blackbox to help you code faster

1 from rest_framework.status import (HTTP_200_OK, HTTP_201_CREATED,

2 HTTP_202_ACCEPTED, HTTP_400_BAD_REQUEST,

3 HTTP_401_UNAUTHORIZED, HTTP_403_FORBIDDEN,

4 HTTP_404_NOT_FOUND, HTTP_409_CONFLICT,

5 HTTP_429_TOO_MANY_REQUESTS,

6 HTTP_500_INTERNAL_SERVER_ERROR,

7 HTTP_502_BAD_GATEWAY,

8 HTTP_503_SERVICE_UNAVAILABLE)

9

10

11 class STATUS_CODES:

12 OK = HTTP_200_OK

13 CREATED = HTTP_201_CREATED

14 ACCEPTED = HTTP_202_ACCEPTED

15 BAD_REQ = HTTP_400_BAD_REQUEST

16 NO_AUTH = HTTP_401_UNAUTHORIZED

17 PERMISSION_ERROR = HTTP_403_FORBIDDEN

18 NOT_FOUND = HTTP_404_NOT_FOUND

19 CONFLICT = HTTP_409_CONFLICT

20 RATE_LIMIT = HTTP_429_TOO_MANY_REQUESTS

21 SERVER_ERROR = HTTP_500_INTERNAL_SERVER_ERROR

22 GATEWAY_FAIL = HTTP_502_BAD_GATEWAY

23 UNAVAILABLE = HTTP_503_SERVICE_UNAVAILABLE

Ln 1, Col 1

Spaces: 4

UTF-8

LF

Python

3.12.1 64-bit

Go Live

Blackbox

WorkTimeLine Chart

MONTH	Jan				Feb				Mar				Apr	
WEEK	1	2	3	4	1	2	3	4	1	2	3	4	1	2
Started learning														
Python														
Git & GitHub														
Sql														
Django														
Travel-management-system (Test - Project)														
Django-rest-framework														
Started Project														
Requirement Analysis														
Database design														
Started Backend side														
Documentation														



THANK YOU