МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

**Лабораторная работа № 2**

*по дисциплине «Программирование мобильных устройств»*

Выполнил студен группы ПИбд-32

Емельянов А. С.

Проверил доцент кафедры

«Информационные системы»

Филиппов А.А.

Ульяновск, 2023

**Задание** Разработка экранных форм приложения и настройка навигации.

Необходимо:

1. На основе макетов, полученных в рамках ЛР № 1, разработать экранные формы приложения. При разработке экранных форм можно использовать фрагменты и/или композицию (compose).

2. Настроить навигацию между экранными формами приложения.

3. Разработать необходимые классы-сущности и создать на их основе тестовые наборы данных.

4. Использовать тестовые данные для демонстрации работы приложения.

5. Отчет и изменения проекта загрузить в репозиторий по адресу <http://student.git.athene.tech>

**Решение**

Настроим панель навигации для проекта

Создадим класс для хранения элементов навигации

sealed class NavItem(val route: String, val icon: ImageVector?){  
 object Home : NavItem("home", Icons.Default.*Home*)  
 object Like : NavItem("like", Icons.Default.*Favorite*)  
 object Order : NavItem("order", Icons.Default.*ShoppingCart*)  
 object Profile : NavItem("profile", Icons.Default.*Person*)  
 object SignIn : NavItem("login", null)  
 object SignUp : NavItem("signup", null)  
 object Person : NavItem("person", null)  
 object AdminPanel : NavItem("admin", Icons.Default.*Build*)  
 object AddPanel : NavItem("add", null)  
 object ChangePanel : NavItem("change", null)  
 object AboutSneaker : NavItem("aboutSneaker/{sneakerItem}", null)  
 object ChangeSneaker : NavItem("changeSneaker/{sneakerItem}", null)  
}

Далее для навигации настроим NavController который будет отвечать, за то на какую функцию отрисовки мы хотим вызвать

@Composable  
fun NavController(navController: NavHostController){  
 NavHost(  
 navController = navController,  
 startDestination = NavItem.Home.route  
 )**{** *composable*(NavItem.Home.route)**{** HomeScreen(navController)  
 **}** *composable*(NavItem.Like.route)**{** LikeScreen()  
 **}** *composable*(NavItem.Order.route)**{** OrderScreen()  
 **}** *composable*(NavItem.Profile.route)**{** ProfileScreen(navController)  
 **}** *composable*(NavItem.SignIn.route)**{** LoginScreen()  
 **}** *composable*(NavItem.SignUp.route)**{** SignUpScreen()  
 **}** *composable*(NavItem.Person.route)**{** Person()  
 **}** *composable*(NavItem.AdminPanel.route)**{** AdminPanel(navController)  
 **}** *composable*(NavItem.AddPanel.route)**{** AddPanel()  
 **}** *composable*(NavItem.ChangePanel.route)**{** ChangePanel(navController)  
 **}** *composable*(NavItem.AboutSneaker.route) **{** backStackEntry **->** val sneakerItemString = backStackEntry.arguments?.getString("sneakerItem")  
 val sneakerItem = Gson().fromJson(sneakerItemString, SneakerItem::class.*java*)  
 sneakerItem?.*let* **{** AboutSneaker(**it**) **}  
 }** *composable*(NavItem.ChangeSneaker.route) **{** backStackEntry **->** val sneakerItemString = backStackEntry.arguments?.getString("sneakerItem")  
 val sneakerItem = Gson().fromJson(sneakerItemString, SneakerItem::class.*java*)  
 sneakerItem?.*let* **{** ChangeSneaker(**it**) **}  
 }  
 }**}

И сама функция отрисовки нижней панели навигации

@SuppressLint("UnusedMaterialScaffoldPaddingParameter")  
@Composable  
fun Navigate(){  
 val navController = rememberNavController()  
 val listItem = *listOf*(  
 NavItem.Home,  
 NavItem.Like,  
 NavItem.Order,  
 NavItem.Profile,  
 NavItem.AdminPanel,  
 )  
  
 Scaffold(bottomBar = **{** BottomNavigation(  
 backgroundColor = Color.White  
 ) **{** val navBackStackEntry = navController.currentBackStackEntryAsState()  
 val currentState = navBackStackEntry.value  
  
 listItem.*forEach* **{** it **->** val isSelected = currentState?.destination?.route == it.route  
  
 BottomNavigationItem(  
 selected = isSelected,  
 onClick = **{** if(!isSelected){  
 navController.graph.startDestinationRoute?.*let* **{** navController.popBackStack(**it**, inclusive = true)  
 **}** navController.navigate(it.route)**{** launchSingleTop  
 **}** }  
 navController.navigate(it.route)  
  
 **}**,  
 icon = **{** val iconModifier = if (isSelected) {  
 Modifier  
 .*background*(color = colorResource(id = R.color.*figma\_blue*), shape = *CircleShape*)  
 .*padding*(8.*dp*)  
 } else {  
 Modifier  
 }  
  
 it.icon?.*let* **{** it1 **->** Icon(  
 imageVector = it1,  
 contentDescription = null,  
 modifier = iconModifier.then(Modifier.*size*(24.*dp*))  
 )  
 **}  
 }** )  
 **}  
 }  
 }**) **{** NavController(navController = navController)  
 **}**}

Нижняя панель выглядит следующим образом



Далее создадим header который будет висеть на каждой странице

@Composable  
fun Header() {  
 Box(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*background*(Color.White)  
 ) **{** Image(  
 painter = painterResource(id = R.drawable.*logo*),  
 contentDescription = "image",  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*padding*(0.*dp*, 10.*dp*, 0.*dp*, 0.*dp*)  
 )  
 **}**}



Чтобы это все работало и находилось на экране, вызовем их в классе MainActivity

class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *setContent* **{** MainContent()  
 **}** }  
}  
  
@Composable  
fun MainContent() {  
 Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 ) **{** Header()  
 Navigate()  
 **}**}

Далее перейдем к созданию самих экранных форм согласно макету

Для разбиения отдельных компонентов будет выносить их в разные функции

**Главная страница**

Начнем сверху вниз и создадим поле для ввода текста для поиска по каталогу

@Composable  
fun SearchField(  
 modifier: Modifier = Modifier,  
 onSearch: (String) -> Unit  
) {  
 var searchText by remember **{** *mutableStateOf*("") **}** Box(  
 modifier = modifier  
 .*fillMaxWidth*()  
 .*background*(colorResource(id = R.color.*figma*))  
 .*padding*(horizontal = 16.*dp*, vertical = 8.*dp*)  
 ) **{** Row(  
 modifier = Modifier  
 .*fillMaxWidth*(),  
 verticalAlignment = Alignment.CenterVertically,  
 content = **{** BasicTextField(  
 value = searchText,  
 onValueChange = **{** searchText = **it** onSearch(**it**)  
 **}**,  
 textStyle = TextStyle(color = Color.Black),  
 modifier = Modifier  
 .*weight*(1f)  
 )  
  
 SearchIcon(  
 imageVector = Icons.Default.*Search*,  
 contentDescription = "Search Icon",  
 modifier = Modifier.*size*(24.*dp*)  
 )  
 **}** )  
 **}**}



Далее создадим элементы отвечающие за сортировку по определенным брендам

Создадим класс который будет хранить в себе пока что id картинки, далее можно будет его расширить

data class ItemFilterByBrand(  
 val imageId: Int,  
)

Создадим врестку для этого элемента

@Composable  
fun ItemRow(item: ItemFilterByBrand) {  
 Column(  
 horizontalAlignment = Alignment.CenterHorizontally,  
 modifier = Modifier  
 .*clip*(*RoundedCornerShape*(10.*dp*))  
 .*background*(colorResource(id = R.color.*figma*))  
  
 ) **{** Image(  
 painter = painterResource(id = item.imageId),  
 contentDescription = "image",  
 contentScale = ContentScale.Crop,  
 modifier = Modifier  
 .*padding*(13.*dp*, 3.*dp*, 13.*dp*, 3.*dp*)  
 .*size*(30.*dp*)  
 )  
 **}**}

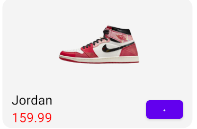
Выглядит он у нас вот так



Далее создадим саму ленту с каталогом

Карточка для кроссовка

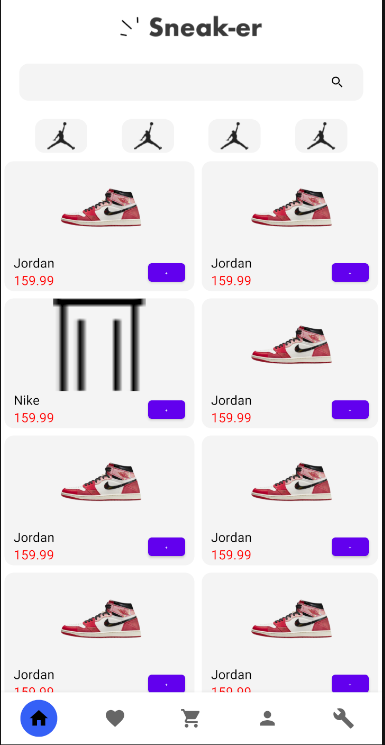
@Composable  
fun CardSneaker(item: SneakerItem, navController: NavHostController) {  
 val maxWidth = (*LocalConfiguration*.current.screenWidthDp / 2).*dp* Box(  
 modifier = Modifier  
 .*padding*(4.*dp*)  
 .*widthIn*(maxWidth)  
 .*clickable* **{** val sneakerItemString = Gson().toJson(item)  
 navController.navigate("aboutSneaker/${sneakerItemString}")  
 **}** ) **{** Column(  
 horizontalAlignment = Alignment.CenterHorizontally,  
 modifier = Modifier  
 .*clip*(*RoundedCornerShape*(10.*dp*))  
 .*background*(colorResource(id = R.color.*figma*))  
 .*widthIn*(maxWidth)  
 ) **{** Image(  
 painter = painterResource(id = item.imageId),  
 contentDescription = "image",  
 contentScale = ContentScale.FillWidth,  
 modifier = Modifier  
 .*size*(100.*dp*)  
 )  
  
 Column **{** Row(  
 modifier = Modifier  
 .*widthIn*(maxWidth),  
 horizontalArrangement = Arrangement.SpaceBetween,  
 ) **{** Column(  
 horizontalAlignment = Alignment.Start,  
 modifier = Modifier  
 .*padding*(10.*dp*, 0.*dp*)  
 ) **{** item.name?.*let* **{** Text(text = **it**) **}** Text(text = item.price.toString(), color = Color.Red)  
 **}** Column(  
 modifier = Modifier  
 .*padding*(10.*dp*)  
 ) **{** Button(  
 onClick = **{** /\**TODO*\*/ **}**,  
 modifier = Modifier  
 .*size*(40.*dp*, 20.*dp*)  
 ) **{** Text(text = "+", fontSize = 4.*sp*)  
 **}  
 }  
 }  
 }  
 }  
 }**}



Теперь будем вызывать все эти функции в одной главной для страницы

@Composable  
fun HomeScreen(navHostController: NavHostController) {  
 Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*background*(Color.White)  
 ) **{** Row **{** // Поле для поиска  
 SearchField(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(20.*dp*)  
 .*clip*(*RoundedCornerShape*(10.*dp*))  
 ) **{** searchText **->** // Обработка введенного текста поиска  
 **}  
 }** Row **{** LazyRow(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(0.*dp*, 0.*dp*, 0.*dp*, 5.*dp*),  
  
 horizontalArrangement = Arrangement.SpaceEvenly  
 )**{** *itemsIndexed*(  
 *listOf*(  
 ItemFilterByBrand(R.drawable.*jordan*),  
 ItemFilterByBrand(R.drawable.*jordan*),  
 ItemFilterByBrand(R.drawable.*jordan*),  
 ItemFilterByBrand(R.drawable.*jordan*)  
 )  
 )**{** \_, item**->** ItemRow(item = item)  
 **}  
 }  
 }** Row **{** LazyColumn(  
 modifier = Modifier  
 .*fillMaxSize*()  
 ) **{** val list = *getSneakers*()  
 val numColumns = 2  
 val chunkedList = list.*chunked*(numColumns)  
  
 *itemsIndexed*(chunkedList) **{** \_, chunkedListItem **->** Row(  
 horizontalArrangement = Arrangement.Center,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 ) **{** for (item in chunkedListItem) {  
 CardSneaker(item, navHostController)  
 }  
 **}  
 }  
 }  
 }  
  
 }**}

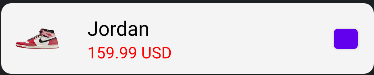
В целом страница будет выглядеть так



**Страница с избранным**

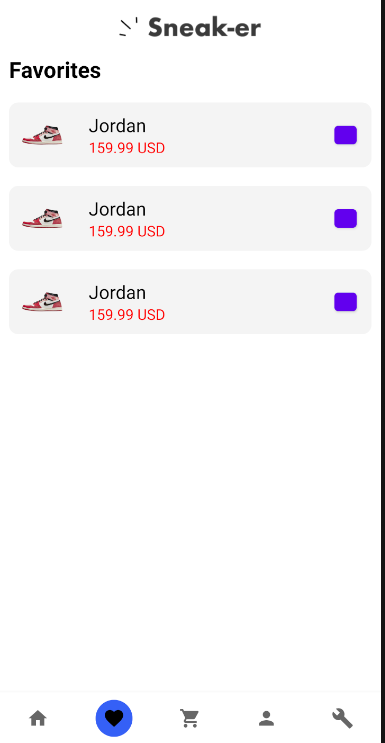
Оставим тот же класс кроссовка что у нас и был, немного переделаем макет карточки товара

@Composable  
fun CardSneakerLike(item: SneakerItem) {  
 Row(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(10.*dp*)  
 .*clip*(*RoundedCornerShape*(10.*dp*))  
 .*background*(colorResource(id = R.color.*figma*)),  
 verticalAlignment = Alignment.CenterVertically,  
 horizontalArrangement = Arrangement.SpaceBetween  
 ) **{** Image(  
 painter = painterResource(id = item.imageId),  
 contentDescription = "image",  
 contentScale = ContentScale.FillWidth,  
 modifier = Modifier  
 .*size*(70.*dp*)  
 .*padding*(10.*dp*)  
 .*clip*(*RoundedCornerShape*(10.*dp*))  
 )  
  
 Column(  
 modifier = Modifier  
 .*weight*(1f)  
 .*padding*(start = 16.*dp*)  
 ) **{** item.name?.*let* **{** Text(text = **it**, fontSize = 20.*sp*) **}** Text(text = "${item.price} USD", color = Color.Red, fontSize = 16.*sp*)  
 **}** Button(  
 onClick = **{** /\**TODO*\*/ **}**,  
 modifier = Modifier  
 .*size*(40.*dp*, 20.*dp*)  
 .*padding*(end = 16.*dp*)  
 ) **{** Text(text = "+", fontSize = 4.*sp*)  
 **}  
 }**}



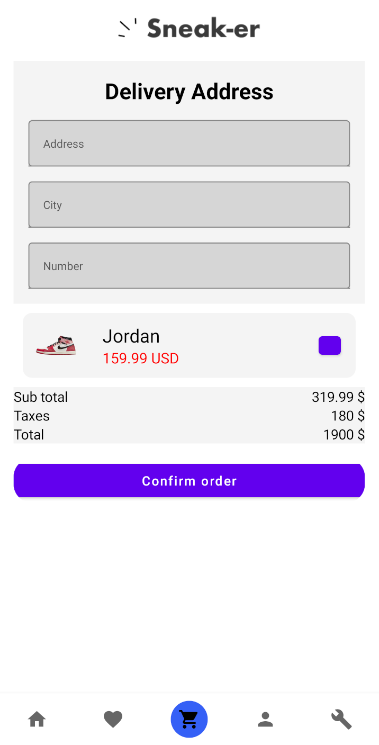
И выведем все это на странице с избранным

@Composable  
fun LikeScreen() {  
 Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*background*(Color.White)  
 )**{** Text(  
 text = "Favorites",  
 style = MaterialTheme.typography.h5,  
 fontWeight = FontWeight.Bold,  
 modifier = Modifier  
 .*padding*(10.*dp*, 10.*dp*)  
 )  
 Row **{** LazyColumn(  
 modifier = Modifier  
 .*fillMaxSize*()  
 ) **{** *itemsIndexed*(  
 *listOf*(  
 SneakerItem(R.drawable.*sneaker*, "Jordan", 159.99),  
 SneakerItem(R.drawable.*sneaker*, "Jordan", 159.99),  
 SneakerItem(R.drawable.*sneaker*, "Jordan", 159.99),  
 )  
 )**{**\_, item**->** CardSneakerLike(item = item)  
  
 **}  
 }  
 }  
 }**}



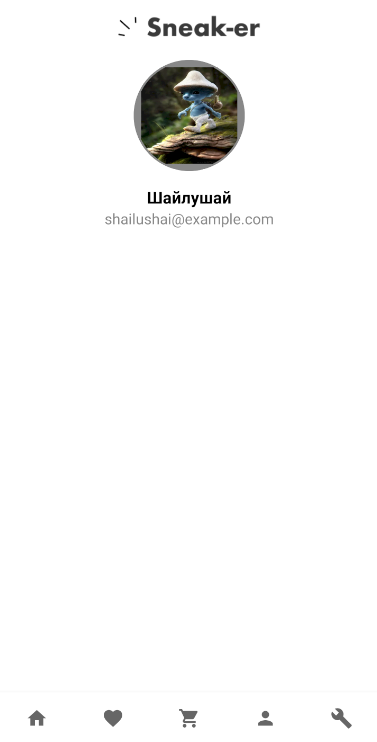
**Страница с оформлением заказа**

@Composable  
fun OrderScreen() {  
  
 var address by remember **{** *mutableStateOf*("") **}** var city by remember **{** *mutableStateOf*("") **}** var number by remember **{** *mutableStateOf*("") **}** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*background*(Color.White)  
 .*padding*(16.*dp*),  
 )**{** Row(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*background*(colorResource(id = R.color.*figma*)),  
 horizontalArrangement = Arrangement.Center  
 ) **{** Column(  
 modifier = Modifier  
 .*padding*(16.*dp*),  
 horizontalAlignment = Alignment.CenterHorizontally  
  
 ) **{** Text(text = "Delivery Address", fontSize = 24.*sp*, fontWeight = FontWeight.Bold)  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = address,  
 onValueChange = **{** address = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*)),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Address",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = city,  
 onValueChange = **{** city = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*)),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "City",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = number,  
 onValueChange = **{** number = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*)),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Number",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
 **}  
 }** Row **{** LazyColumn **{** *itemsIndexed*(  
 *listOf*(  
 SneakerItem(R.drawable.*sneaker*, "Jordan", 159.99),  
 )  
 )**{**\_, item**->** CardSneakerLike(item = item)  
 **}  
 }  
 }** Row(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*background*(colorResource(id = R.color.*figma*)),  
 horizontalArrangement = Arrangement.SpaceBetween  
 )**{** Row(  
 modifier = Modifier.*weight*(1f),  
 horizontalArrangement = Arrangement.Start  
 )**{** Text(text = "Sub total", fontSize = 15.*sp*)  
 **}** Row(  
 modifier = Modifier.*weight*(1f),  
 horizontalArrangement = Arrangement.End  
 )**{** Text(text = "319.99 $", fontSize = 15.*sp*)  
 **}  
 }** Row(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*background*(colorResource(id = R.color.*figma*)),  
 horizontalArrangement = Arrangement.SpaceBetween  
 )**{** Row(  
 modifier = Modifier.*weight*(1f),  
 horizontalArrangement = Arrangement.Start  
 )**{** Text(text = "Taxes", fontSize = 15.*sp*)  
 **}** Row(  
 modifier = Modifier.*weight*(1f),  
 horizontalArrangement = Arrangement.End  
 )**{** Text(text = "180 $", fontSize = 15.*sp*)  
 **}  
 }** Row(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*background*(colorResource(id = R.color.*figma*)),  
 horizontalArrangement = Arrangement.SpaceBetween  
 )**{** Row(  
 modifier = Modifier.*weight*(1f),  
 horizontalArrangement = Arrangement.Start  
 )**{** Text(text = "Total", fontSize = 15.*sp*)  
 **}** Row(  
 modifier = Modifier.*weight*(1f),  
 horizontalArrangement = Arrangement.End  
 )**{** Text(text = "1900 $", fontSize = 15.*sp*)  
 **}  
 }** Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 Button(  
 onClick = **{  
  
 }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*clip*(*RoundedCornerShape*(20.*dp*))  
 ) **{** Text("Confirm order")  
 **}  
 }**}



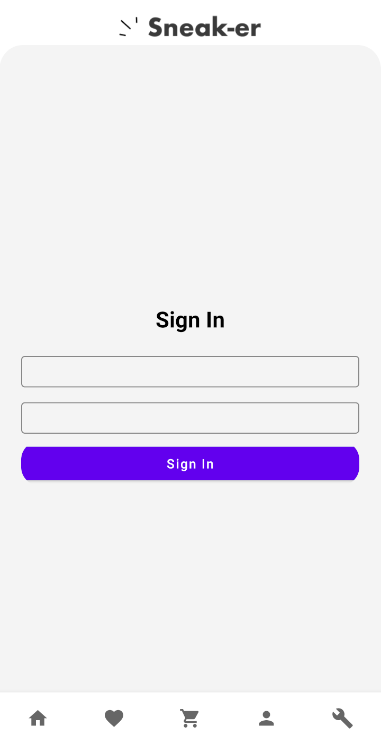
**Страница профиля**

@Composable  
fun Person() {  
 Column(  
 modifier = Modifier  
 .*background*(Color.White)  
 .*fillMaxSize*()  
 .*padding*(16.*dp*),  
 verticalArrangement = Arrangement.Top,  
 horizontalAlignment = Alignment.CenterHorizontally  
 ) **{** Image(  
 contentScale = ContentScale.FillBounds,  
 painter = painterResource(id = R.drawable.*shailushai*),  
 contentDescription = null,  
 modifier = Modifier  
 .*size*(120.*dp*)  
 .*clip*(*CircleShape*)  
 .*border*(2.*dp*, Color.Gray, *CircleShape*)  
 .*background*(Color.Gray)  
 .*padding*(8.*dp*)  
 )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 Text(  
 text = "Шайлушай",  
 fontSize = 18.*sp*,  
 fontWeight = FontWeight.Bold  
 )  
  
 Text(  
 text = "shailushai@example.com",  
 fontSize = 16.*sp*,  
 color = Color.Gray  
 )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 **}**}



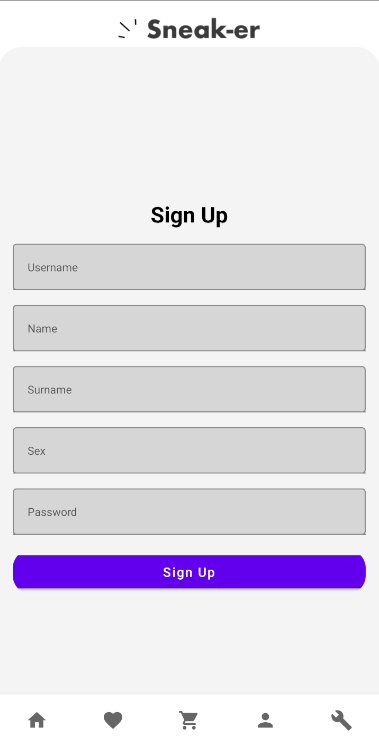
**Страница входа**

@Composable  
fun LoginScreen() {  
 var username by remember **{** *mutableStateOf*("") **}** var password by remember **{** *mutableStateOf*("") **}** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*clip*(*RoundedCornerShape*(25.*dp*))  
 .*background*(colorResource(id = R.color.*figma*))  
 .*padding*(15.*dp*, 0.*dp*)  
 ,  
 verticalArrangement = Arrangement.Center,  
 horizontalAlignment = Alignment.CenterHorizontally  
 ) **{** Text1(text = "Sign In", fontSize = 24.*sp*, fontWeight = FontWeight.Bold)  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 BasicTextField(  
 value = username,  
 onValueChange = **{** username = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*padding*(8.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(8.*dp*),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 )  
  
 BasicTextField(  
 value = password,  
 onValueChange = **{** password = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*padding*(8.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(8.*dp*),  
 singleLine = true,  
 visualTransformation = PasswordVisualTransformation(),  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 )**{  
  
 }** Button(  
 onClick = **{  
  
 }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(8.*dp*, 0.*dp*, 8.*dp*, 0.*dp*)  
 .*clip*(*RoundedCornerShape*(20.*dp*))  
 ) **{** Text1("Sign In")  
 **}  
 }**}



**Страница регистрации**

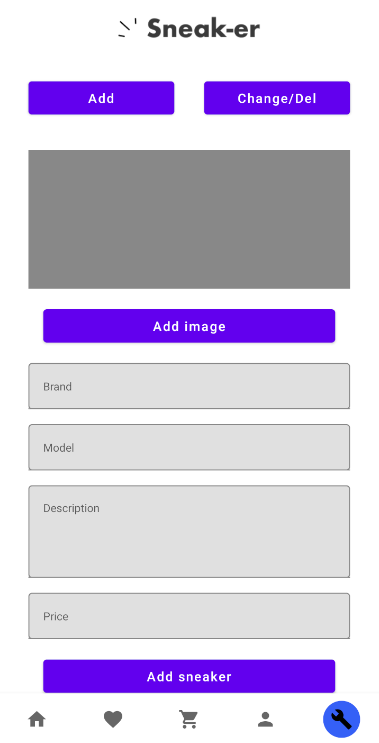
@Composable  
fun SignUpScreen() {  
 var username by remember **{** *mutableStateOf*("") **}** var password by remember **{** *mutableStateOf*("") **}** var sex by remember **{** *mutableStateOf*("") **}** var name by remember **{** *mutableStateOf*("") **}** var surname by remember **{** *mutableStateOf*("") **}** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*clip*(*RoundedCornerShape*(25.*dp*))  
 .*background*(colorResource(id = R.color.*figma*))  
 .*padding*(15.*dp*, 0.*dp*)  
 ,  
 verticalArrangement = Arrangement.Center,  
 horizontalAlignment = Alignment.CenterHorizontally  
 ) **{** Text(text = "Sign Up", fontSize = 24.*sp*, fontWeight = FontWeight.Bold)  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = username,  
 onValueChange = **{** username = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Username",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = name,  
 onValueChange = **{** name = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*)),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Name",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = surname,  
 onValueChange = **{** surname = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Surname",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = sex,  
 onValueChange = **{** sex = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Sex",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = password,  
 onValueChange = **{** password = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 singleLine = true,  
 visualTransformation = PasswordVisualTransformation(),  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Password",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 Button(  
 onClick = **{  
  
 }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*clip*(*RoundedCornerShape*(20.*dp*))  
 ) **{** Text("Sign Up")  
 **}  
 }**}



**Страница админа**

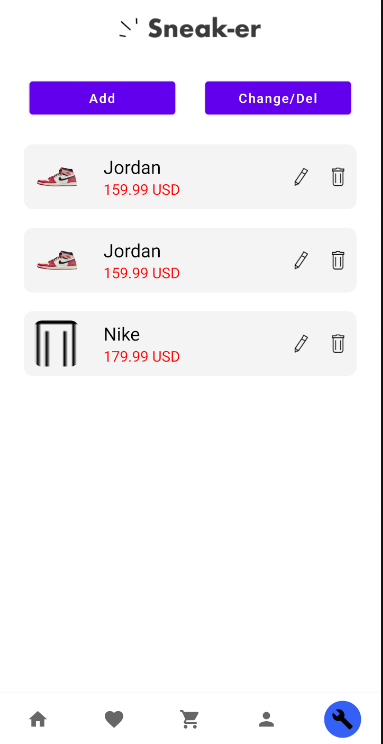
**Экран добавления товара**

@Composable  
@Preview  
fun AddPanel(){  
 var brand by remember **{** *mutableStateOf*("") **}** var model by remember **{** *mutableStateOf*("") **}** var description by remember **{** *mutableStateOf*("") **}** var price by remember **{** *mutableStateOf*("") **}** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*background*(Color.White)  
 .*padding*(16.*dp*)  
 )**{** Box(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(150.*dp*)  
 .*background*(Color.Gray)  
 ) **{  
 }** Button(  
 onClick = **{  
 }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*clip*(*RoundedCornerShape*(20.*dp*))  
 .*padding*(16.*dp*)  
 ) **{** Text("Add image")  
 **}** TextField(  
 value = brand,  
 onValueChange = **{** brand = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Brand",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = model,  
 onValueChange = **{** model = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Model",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = description,  
 onValueChange = **{** description = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(100.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Description",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = price,  
 onValueChange = **{** price = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Price",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Button(  
 onClick = **{  
  
 }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*clip*(*RoundedCornerShape*(20.*dp*))  
 .*padding*(16.*dp*)  
 ) **{** Text("Add sneaker")  
 **}  
 }**}



**Страница с редактирование и удалением товара**

@Composable  
fun ChangePanel(navHostController: NavHostController) {  
 Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*background*(Color.White)  
 )**{** Row **{** LazyColumn(  
 modifier = Modifier  
 .*fillMaxSize*()  
 ) **{** *itemsIndexed*(  
 *listOf*(  
 SneakerItem(R.drawable.*sneaker*, "Jordan", 159.99),  
 SneakerItem(R.drawable.*sneaker*, "Jordan", 159.99),  
 SneakerItem(R.drawable.*trash*, "Nike", 179.99),  
 )  
 )**{**\_, item**->** CardSneakerForChange(item = item, navHostController)  
  
 **}  
 }  
 }  
 }**}



**Страница с редактированием товара**

@Composable  
fun ChangeSneaker(sneaker: SneakerItem) {  
 var brand by remember **{** *mutableStateOf*("") **}** var model by remember **{** *mutableStateOf*("") **}** var description by remember **{** *mutableStateOf*("") **}** var price by remember **{** *mutableStateOf*("") **}** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*background*(Color.White)  
 .*padding*(16.*dp*)  
 )**{** Image(  
 painter = painterResource(id = sneaker.imageId),  
 contentDescription = "image",  
 contentScale = ContentScale.FillWidth,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(200.*dp*)  
 )  
 Button(  
 onClick = **{  
 }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*clip*(*RoundedCornerShape*(20.*dp*))  
 .*padding*(16.*dp*)  
 ) **{** Text("Change image")  
 **}** TextField(  
 value = sneaker.name.*toString*(),  
 onValueChange = **{** brand = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** )  
 )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = model,  
 onValueChange = **{** model = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Model",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = description,  
 onValueChange = **{** description = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(100.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** ),  
 placeholder = **{** Text(  
 text = "Description",  
 style = TextStyle(fontSize = 12.*sp*)  
 )  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 TextField(  
 value = sneaker.price.toString(),  
 onValueChange = **{** price = **it }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*)  
 .*border*(1.*dp*, Color.Gray, *RoundedCornerShape*(4.*dp*))  
 .*padding*(0.*dp*),  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 imeAction = ImeAction.Next  
 ),  
 keyboardActions = KeyboardActions(  
 onNext = **{  
  
 }** )  
 )  
  
 Button(  
 onClick = **{  
  
 }**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*clip*(*RoundedCornerShape*(20.*dp*))  
 .*padding*(16.*dp*)  
 ) **{** Text("Save")  
 **}  
 }**}

