Модуль пользовательского интерфейса
- App(): ReactElement
- NavigationEntry(): ReactElement
- HorizontalPicker(props: HorizontalPickerProps): ReactElement
- TextInput(props: TextInputProps): ReactElement
- LoginScreen(): ReactElement
- RegisterScreen(): ReactElement
- HistoryScreen(): ReactElement
- HomeDrawer(): ReactElement
- PaymentMethodComponent(props: PMProps): ReactElement
- AddCard(): ReactElement
- PaymentsList(): ReactElement
- ProfileScreen(): ReactElement
- Status(): ReactElement
- Pointer(): ReactElement
- HomeScreen(): ReactElement
- RideRequest(): ReactElement
- DriverOnRideStatus(): ReactElement
- SearchBlock(): ReactElement
- SearchResultsBlock(): ReactElement
- ChooseClass(): ReactElement
- CustomerRideStatus(): ReactElement

Модуль мобильной бизнес-логики
- appSaga()
- initializationSaga()
- loginSaga(action: Action)
- listenForLogin()
- logoutSaga()
- listenForLogout()
- registerSaga(action: Action)
- listenForRegister()
- fetchHistorySaga()
- listenForFetchHistory()
- getUserSaga()
- getPaymentMethods()
- listenForGetPaymentMethods()
- removePaymentMethodSaga(action: Action)
- listenForRemovePaymentMethod()
- setAsDefaultPaymentMethodSaga(action: Action)
- listenForSetAsDefaultPaymentMethod()
- addCardSaga(action: Action)
- listenForAddCard()
- paySaga(action: Action)
- listenForPay()
- inititializeMapSaga()
- listenForInitializeMap()
- receiveLocationUpdateSaga(action: Action)
- bootstrapGPSSubscription()
- receiveWebSocketMessageSaga(action: Action)
- bootstrapWebSocketSubscription()
- chooseRouteSaga()
- listenForChooseRoute()
- fetchPlacesSaga()
- listenForFetchPlaces()
- prepareRideDataSaga(action: Action)
- listenForPrepareRideData()
- requestRideSaga(action: Action)
- listenForRequestRide()
- setChosenLocationSaga(action: Action)
- listenFirSetChosenLocation()
- setRouteLocationSaga(action: Action)
- listenForSetRouteLocation()

RestGatewayAPI
- -authToken: string
- -axios: AxiosInstance
- +initialize()
- +setAuthToken(token: string): AxiosInstance
- +post<K, T>(route: string, data: K): Promise<T>>
- +get<K, T>(route: string, config: K): Promise<Response<T>>

ConnectionGatewayAPI
- -authToken: string
- -isClient: boolean
- -listeners: Array<WSMessage => void>
- -socket: Socket | null
- +setIsClient(isClient: boolean)
- +setAuthToken(token: string)
- +addEventListener(listener: WSMessage => void): () => void
- +connect()
- +send<T>(event: WSMessageType, data: T)
- +disconnect()
- -retryConnection()

PaymentsAPI
- -restGatewayAPI: RestGatewayAPI
- +getPaymentMethods(): Promise<Response<Array<PaymentMethod>>>
- +setAsDefaultMethod(id: number): Promise<Response<unknown>>
- +addCard(data: CardMethodDetails): Promise<Response<unknown>>
- +createPaymentIntent(data: CreatePaymentIntentInput): Promise<Response<string>>
- +removePaymentMethod(id: number): Promise<Response<unknown>>
- +paymentFinished(data: PaymentFinishedInput): Promise<Response<unknown>>

NavigationService
- -navigationRef: NavigationContainerRef<any> | null
- +setNavigationRef(ref: NavigationContainerRef<any>)
- +goBack()

MapService
- -map: MutableRefObject<MapView>
- +getMapRef(): MutableRefObject<MapView>
- +animateCamera(location: Location, zoom: number)
- +animateToRegion(pos1: Location, pos2: Location)

GeolocationService
- +latestLocation: Location
- +initialize()
- +getLocation(): Location
- +subscribeToPositionChange(Location => void): () => void

Модуль работы с хранилищем
- userReducer(state: UserState, action: Action<any>): UserState
- historyReducer(state: HistoryState, action: Action<any>): HistoryState
- paymentsReducer(state: PaymentsState, action: Action<any>): PaymentsState
- homeReducer(state: HomeState, action: Action<any>): HomeState
- rootReducer(state: ApplicationState, action: Action<any>): ApplicationState

HomeAPI
- -restGatewayAPI: RestGatewayAPI
- -connectionGatewayAPI: ConnectionGatewayAPI
- +decode(data: Location): Promise<Response<ExtendedLocation>>
- +updateMyLocation(data: Location): Promise<Response<T>>
- +requestRide(data: ExtendedRequestRide)
- +answerToRideRequest(answer: WSMessageType)
- +fetchPlaces(toSearch: string): Promise<Response<Array<ExtendedLocation>>>
- +calculateRideData(from: ExtendedLocation, to: ExtendedLocation): Promise<Response<RideRequest>>
- +declineRideRequest()
- +updateRideStatus(status: RideStatus)

AuthAPI
- -restGatewayAPI: RestGatewayAPI
- +login(email: string, password: string): Promise<Response<Tokens>>
- +register(data: RegisterPayload): Promise<Response<Tokens>>
- +refreshTokens<K, T>(token: string): Promise<Response<Tokens>>

ProfileAPI
- -restGatewayAPI: RestGatewayAPI
- +getUser(): Promise<Response<User>>>

HistoryAPI
- -restGatewayAPI: RestGatewayAPI
- +getHistory(): Promise<Response<Array<Ride>>>

<<enumeration>> WSMessageType
- RideRequest
- RideStatusChange
- RideAccept
- RideStopSearch
- RideTimeout
- LocationUpdate

Tokens
- +token: string
- +refreshToken: string

ExtendedRideRequest
- +calculatedTime: number
- +route: Array<Location>
- +to: ExtendedLocation
- +from: ExtendedLocation
- +cost: number
- +carClass: CarClass

<<enumeration>> CardBrand
- Visa
- Mastercard
- AmericanExpress

Driver
- +id: number
- +carBrand: string
- +carClass: CarClass
- +balance: number

<<enumeration>> CarClass
- Economy
- Comfort
- Business

Ride
- +id: number
- +client: User
- +driver: User
- +const: number
- +payment: Optional<Payment>
- +startTime: number
- +endTime: number
- +to: string
- +from: string
- +status: RideStatus
- +paid: boolean

<<enumeration>> RideStatus
- Starting
- InProgress
- Completed
- NoRide

User
- +id: number
- +email: string
- +phone: string
- +firstName: string
- +lastName: string
- +gender: Gender
- +driver: Optional<Driver>

<<enumeration>> PaymentMethodType
- Cash
- Card

Location
- +latitude: number
- +longitude: number

<<enumeration>> Gender
- Male
- Female

ExtendedLocation
- +latitude: number
- +longitude: number
- +readableDescription: string

RideRequest
- +calculatedTime: number
- +route: Array<Location>
- +classes: Record<CarClass, number>

PaymentMethod
- +id: number
- +type: PaymentMethodType
- +isDefault: boolean
- +details: Optional<PaymentMethodDetails>

PaymentMethodDetails
- +id: number
- +lastFour: string
- +exp: string
- +holder: string
- +brand: CardBrand
- +stripePaymentId: string

AuthController
- -authService: AuthService
- +constructor(authService: AuthService)
- +login(data: LoginInput): Promise<Tokens>
- +register(data: RegisterInput): Promise<Tokens>
- +refreshTokens(data: RefreshInput): Promise<Tokens>

UserController
- -userService: UserService
- +constructor(userService: UserService)
- +getUser(req: Request): Promise<User>
- +getHistory(req: Request): Promise<Array<Ride>>
- +getCurrentStatus(eq: Request): Promise<Ride | null>

PaymentController
- -paymentService: PaymentService
- +constructor(paymentService: PaymentService)
- +getPaymentMethods(): Promise<Array<PaymentMethod>>
- +setDefaultMethod(req: Request, data: SetAsDefaultOrRemoveInput): Promise<StatusWrapper>
- +addCard(req: Request, data: AddCardInput): Promise<StatusWrapper>
- +createPaymentIntent(req: Request, data: CreatePaymentIntentInput): Promise<CreatePaymentIntentOutput>
- +confirmPayment(req: Request, data: ConfirmPaymentInput): Promise<StatusWrapper>
- +removePaymentMethod(req: Request, data: SetAsDefaultOrRemoveInput): Promise<StatusWrapper>

MapsController
- -mapsService: MapsService
- +constructor(mapsService: MapsService)
- +decode(data: DecodeInput): Promise<DecodeOutput>
- +direction(data: DirectionInput): Promise<DirectionOutput>
- +places(data: PlacesInput): Promise<PlacesOutput>

RTCController
- -rtcService: RTCService
- +constructor(rtcService: RTCService)
- +handleConnection(socket: Socket)
- +handleDisconnect(socket: Socket)
- +handleLocationUpdate(data: WithUserRole<Location>, socket: Socket)
- +handleRideRequest(data: WithUserRole<ExtendedRideRequest>, socket: Socket)
- +handleRideStopSearch(socket: Socket)
- +handleRideStatusChange(data: WithUserRole<RideStatus>, socket: Socket)

AuthService
- -authDataRepository: AuthDataRepository
- -tokenProvider: TokenProvider
- -userService: UserService
- -hasher: Hasher
- +constructor(authService: AuthService, tokenProvider: TokenProvider, userService: UserService, hasher: Hasher)
- +login(email: string, password: string): Promise<Tokens>
- +register(data: RegisterInput): Promise<Tokens>
- +refreshTokens(data: RefreshInput): Promise<Tokens>

UserService
- -userRepository: UserRepository
- -driverRepository: DriverRepository
- -paymentService: PaymentService
- -rideRepository: RideRepository
- +constructor(userRepository: UserRepository, driverRepository: DriverRepository, paymentService: PaymentService, rideRepository: RideRepository)
- +getUser(id: number): Promise<User>
- +createUser(data: RegisterInput): Promise<User>
- +getHistory(id: number): Promise<Array<Ride>>
- +getLatestRide(id: number): Promise<Ride | null>

PaymentService
- -paymentRepository: PaymentRepository
- -paymentMethodRepository: PaymentMethodRepository
- -paymentMethodDetailsRepository: PaymentMethodDetailsRepository
- -userRepository: UserRepository
- -rideRepository: RideRepository
- -stripe: Stripe
- +constructor(paymentRepository: PaymentRepository, paymentMethodRepository: PaymentMethodRepository, paymentMethodDetailsRepository: PaymentMethodDetailsRepository, userRepository: UserRepository, rideRepository: RideRepository, stripe: Stripe)
- +getPaymentMethods(): Promise<Array<PaymentMethod>>
- +setDefaultMethod(userId: number, methodId: number)
- +addCard(req: Request, data: AddCardInput)
- +createPaymentIntent(userId: number, data: CreatePaymentIntentInput): Promise<CreatePaymentIntentOutput>
- +confirmPayment(userId: number, data: ConfirmPaymentInput)
- +removePaymentMethod(userId: number, methodId: number)

MapsService
- -geocoding: Geocoding
- -places: Places
- -directions: Directions
- +constructor(geocoding: Geocoding, places: Places, directions: Directions)
- +decode(data: Location): Promise<ExtendedLocation>
- +getDirection(from: Location, to: Location): Promise<DirectionOutput>
- +searchPlaces(toSearch: string): Promise<Array<ExtendedLocation>>

RTCService
- -idBasedSocketHolder: Map<number, Socket>
- -socketBasedIdHolder: WeakMap<Socket, number>
- -clientDataHolder: Map<number, SocketUserInfo>
- -driverDataHolder: Map<number, DriverSocketUserInfo>
- -tokenProvider: TokenProvider
- -userService: UserService
- -geoUtils: GeoUtils
- -rideRepository: RideRepository
- +constructor(tokenProvider: TokenProvider, userService: UserService, geoUtils: GeoUtils, rideRepository: RideRepository)
- +getUserFromSocket(socket: Socket): Promise<User>
- +addUser(socket: Socket)
- +removeUser(socket: Socket)
- +handleLocationUpdate(data: WithUserRole<Location>, socket: Socket)
- +handleRideRequest(data: WithUserRole<ExtendedRideRequest>, socket: Socket)
- +handleRideStopSearch(socket: Socket)
- +handleRideStatusChange(data: WithUserRole<RideStatus>, socket: Socket)

Repository

AuthDataRepository

UserRepository

DriverRepository

RideRepository

PaymentRepository

PaymentMethodRepository

PaymentMethodDetailsRepository

Stripe
- -stripeSecretKey: string
- -usdRate: number
- -stripe: StripeSDK
- +createIntent(email: string, amount: number, requestThreeDSecure: boolean, cusromerId: Optional<number>): Promise<CreateIntentOutput>

GoogleMaps
- +client: Client

Directions
- -maps: GoogleMaps
- -mockDirections: string
- +constructor(maps: GoogleMaps)
- +getDirection(from: Location, to: Location): Promise<Array<Location>>

Geocoding
- -maps: GoogleMaps
- -mockGeocoding: string
- -geoUtils: GeoUtils
- +constructor(geoUtils: GeoUtils, maps: GoogleMaps)
- +decode(data: Location): Promise<ExtendedLocation>

Places
- -maps: GoogleMaps
- -mockPlaces: string
- +constructor(maps: GoogleMaps)
- +search(toSearch: string): Promise<Array<ExtendedLocation>>