MENSAGENS DO ANDROID

MENSAGENS DO PI

APP (Passo a Passo):

1. Actividade de SignUp

1.1. Cria uma conta:

Utilizador insere o Primeiro Nome (pelo menos três letras e não pode ter números), o Último Nome (pelo menos três letras e não pode ter números), o Email (contem os componentes: <username>@<mail-server>.<mail-servertype or server-location>), e a Password (entre 8 a 18 caracteres alfanúmericos, pelo nemos um digito, uma letra minuscula e uma maiuscula).

1.1.1. Se os campos forem inválidos apresenta os respetivos erros e continua na actividade SignUp (1.).

1.1.2. Se os campos forem válidos envia para o PI a seguinte string: "INS\_USER PrimeiroNome, UltimoNome, Email, Password" (notar a separação dos campo com as Virgulas, Brian queres que no inicio da string junte já a string "INS\_USER"? preciso apenas acrescentar acrescentar!) após isto o PI deve verificar se o Email enviado já existe na base de dados:

1.1.2.1. Se o Email já existir na base de dados o PI deve enviar a seguinte string: "KO\_SINGUP" e após isto apresenta o respetivo erro e continua na actividade SignUp (1.).

1.1.2.2. Se o Email não existir na base de dados o PI deve enviar a seguinte string: "OK\_SINGUP" e após isto passo para a acatividade principal (3.).

--> Notar que as strings a retornar é toda em letras maiúsculas.

1.2. Lembra-se das credenciais de Login? então vai para actividade de login (2.).

2. Actividade de Login

2.1. Inicia o login:

Utilizador insere o Email e a Password que são enviados para o PI na seguinte string: "GET\_USER Email, Password" (notar a separação dos campo com as Virgulas, Brian queres que no inicio da string junte alguma string como a string "INS\_USER"?), após isto o PI deve returna a verificação na base de dados das pessoas registadas, ou seja:

2.1.1. o PI deve verificar se o Email enviado (primeiro campo) existe na base de dados,

2.1.1.1. Se o Email existir o PI deve verificar se a Password enviada (segundo campo) é compatível com a password correspondente ao email na base de dados e então passa para o passo (2.1.2.)

2.1.1.2. Se o Email não existir o PI deve returnar a seguinte string: "KO\_EMAIL" após isto é apresentado o respetivo erro e permanece na actividade de Login (2.).

2.1.2. o PI verifica a correspondência da Password

2.1.2.1. Se a Password corresponder com a da base de dados o PI deve returnar a seguinte string: "OK\_LOG" após isto passo para a actividade principal (3.).

2.1.2.2. Se a Password não corresponder o PI deve returnar a seguinte string: "KO\_PASS" após isto é apresentado o respetivo erro e permanece na actividade de Login (2.).

--> Notar que as strings a returnar é toda em letras maiúsculas.

2.2. O utilizador ainda não possui uma conta? então vai para actividade de SingUp (1.).

3. Actividade Principal

3.1. Logo após entrar na actividade principal (após login) o Android envia uma mensagem ao PI na forma: "GET\_STATE PARK" para verificar o estado do parque:

3.1.1. Se o parque está livre o PI envia a string: "FREE" e Android atualiza o estado, ativa o botão da reserva e desativa o botão de chegada.

3.1.2. Se o parque está reservado o PI envia a string: "RESERVED" e Android atualiza o estado, desativa o botão de reserva e o botão de chegada.

3.1.3. Se o parque está ocupado o PI envia a string: "BUSY" e Android atualiza o estado, desativa o botão de reserva e o botão de chegada.

3.1.4. Caso o Android não receba as mensagens pretendidas restabelece o pedido de estado ("STATE") (3.1.).

3.2. Quando o Android aciona o botão de reserva o fragmente de atividade de reserva é aberto:

3.2.1. Se aciona o botão OK, verifica a validade dos dados introduzidos (License Plate tem de estar no formato "CC-NN-NN" ou "NN-CC-NN" ou "NN-NN-CC", e o Standby Time já é predefinido como no mínimo 1 minuto e o utilizador pode escolher entre 1min, 2min ou 3min):

3.2.1.1. Se os dados estão no formato desejado, Android guarda uma flag a dizer que reservou, e envia os dados para o PI na seguinte string: "INS\_BOOKING Email, LicensePlate, StandbyTime" (o Email é para ser mais fácil encontrar o utilizador na base de dados e adicionar as informações de matricula e tempo de reserva), após isto o PI deve dar feedback, ou seja:

3.2.1.1.1. Se o PI enviar uma mensagem com a string "OK\_RESERVE" sei que correu bem então volta para atividade principal (3.) e ativo o botão de chegada e desativo o de reserva.

3.2.1.1.2. Se o PI enviar uma mensagem com a string "KO\_RESERVE" sei que ocorreu erro devolvo uma mensagem e volta ao estado anterior (3.2.).

3.2.1.2. Se os dados não estão no formato desejado é mostrada a mensagem de erro e volta para a actividade principal (3.).

3.2.2. Se aciona o botão CANCEL ou pressiona tecla de goback retorna para a actividade principal (3.) e não faz nada.

3.3. Quando o Android aciona o botão I ARRIVED envia os dados para o PI na seguinte string: "SET\_ARRIVED Email, LicensePlate" (o Email e a matricula é para ter a certeza que a pessoa que chegou foi a que reservou), após isto o PI deve dar feedback, ou seja:

3.3.1. Se o PI enviar uma mensagem com a string "OK\_ARRIVED" sei que ocorreu bem então volta para atividade principal (3.).

3.3.2. Se o PI enviar uma mensagem com a string "KO\_ARRIVED" sei que ocorreu erro devolvo uma mensagem e volta ao estado anterior (3.3.).

(PONTO 3.4. ainda não esté implementado pois lembrei-me deve só depois, mas acho que deve ser implementado e vou faze-lo)

3.4. Quando o Android aciona o botão CANCEL RESERVATION (botão como SingUp) envia para o PI a seguinte string "CANCEL\_BOOKING Email, LicensePlate", após isto o PI deve eliminar os dados da reserva e deve enviar feedback, ou seja:

3.4.1. Se o PI enviar uma mensagem com a string "OK\_CANCEL" sei que ocorreu bem então volta para atividade principal (3.).

3.4.2. Se o PI enviar uma mensagem com a string "KO\_CANCEL" (por alguma razão não conseguiu cancelar a reserva) sei que ocorreu erro devolvo uma mensagem e volta ao estado anterior (3.4.).

3.5. Fora a interação do Android -> PI, deve existir uma interação PI -> Android sempre que o estado do parque é alterado, então:

3.5.1. Se existir um incendio o PI deve enviar uma mensagem com a string: "FIRE"

3.5.2. Se o parque fica livre o PI deve enviar uma mensagem com a string: "FREE"

3.5.3. Se o parque fica ocupado o PI deve enviar uma mensagem com a string: "BUSY"

3.5.4. Se o parque fica reservado o PI deve enviar uma mensagem com a string: "RESERVED"

3.5.5. Se o tempo de reserva expirar para um utilizador o PI deve enviar para esse utilizador a string: "EXPIRED" e também ir para o ponto 3.5.2 (parque livre).

3.5.6. android diz quando a buzina deve parar: "FIRE\_TERMIN PARK"

--> SE ME ESTIVER A FALTAR ALGUM ESTADO DIZ-ME!