Para correr el app: ir al folder de material start, y escribir “ live-server ”
correra en el 8080

Tabla de acciones de Mercurio con sus respectivas instrucciones

Accion	***Instruccion***	***Comentario***
Login	app>src>login>authenticationController.js app>src>login>login.html en authenticationController.js, usar SDK	Ahora mismo: $scope.email $scope.password Ng-model = “email” Ng-model = “password”
Mostrar Drawer	app>src>users>userController.js self.statusMessage self.displayName self.profileImage self.background	Elementos que pueden ser cambiados: Background, profile pic, name, status message.
Editar Drawer	App>src>users>userController.js Hay que guardar los datos del drawer en el DB.	Al clickear profile en el left sidebar.
Mostrar Contactos	Lista de contactos debe estar en un array de objetos -> [{},{},...] dentro de app>src>users>userController.js en var contacts. Sus campos son: name, image, status, phone, address, ringtone, notes, facebook, twitter, crm. Hay un ejemplo escrito ya, en var contacts; sustituyelo por la info real. Lista de contactos debe estar en un array -> [n,n1,n2,...] dentro de app>src>users>userContent.js en var contacts. Este array es llamado por el search async. Hay un ejemplo escrito ya en var contacts, sustituyelo por los nombres reales.	Al clickear Contactos en el left sidebar

Mostrar 1 Contacto	App> src > users > userController.js en contacts hay un ejemplo, sustituir con datos reales. Falta mostrar el logo de crm, en donde estan las redes sociales.	Al clickear un contacto
Editar Contacto	Falta solo guardar los datos en el DB.	Al clickear editar contacto Llamar funciones de deleteContact y addNewContact. La de addNewContact, ya esta implementada. (mirar siguiente 'row')
A~adir nuevo contacto	App > src > users> userController.js Guardar contacto nuevo en el DB, en vez de pushearlo al array. No importa donde lo guardes; programe el fetch, para que los saque en orden alfabetico.	Falta poner feedback de que se a~adio, de que selecciones un numero, por favor, de que el email es valido o no, etc... (Validar)
Dialer	app>src>users>userController.js self.displayName	El nombre del usuario
Phone Settings	app>src>index.html Phone Settings Modal Descomentar el tag de <md-input-container>	Tiene un peque~o bug: La lista de ringtones sale en la parte de atras del modal. (El modal tapa las opciones) Quizas con index-z se resuelva.
Edit Phone Settings	Not yet implemented.	
Account Settings	Not yet implemented.	
Edit Account Settings	Not yet implemented.	
About Mercurio	app>src>leftSidebar.html (Poner un href en el <md-button)	Esto debe ser un link que te envie a un lugar con info. Creo.
Sign Out	app>src>login>authenticationController.js usar SDK	
	app>src>user> userContent.js	

Historial de Llamadas	self.userCallHistory es un array de objetos que contiene: name,time,call,image,type & icon. Name – nombre del usuario pertinente a la llamada Time – hora de la llamada Call – si es recibida, hecha o perdida. Image – foto del usuario pertinente a la llamada type – video-call o phone call(audio call) icon: icono de la llamada: “images/in.png” - entrante, “images/out.png” - saliente, & “images/miss.png” - perdida. Hay un ejemplo escrito ya en self.userCallHistory, sustituyelo por los datos reales.	Ya estan filtradas las llamadas perdidas. Deberia de tener el tiempo de duracion de la lamada y fecha.
Conversaciones	app>src>userProvider.js this.conversations – es el card antes de abrir la conversacion como tal. Array de objetos. Elementos: contact, lastMessage, contactImage, & lastMessageTime. Hay un ejemplo escrito ya en this.conversations, sustituyelo por los datos reales.	
Enviar Notificaciones	app>src>userProvider.js para enviar una notificacion: `var options = { body: "Hi there", icon: "images/mercury.png" }`	

	var notification = new Notification(''Titulo",options); Hay una funcion que se llama sendNotification() - poner codigo ahi.	

To do: Pegar los screen de chats y calls. El chat tiene dos mensajes escritos(hardcoded), uno entrante y uno saliente. Al darle retrieve a los mensajes reales, solo tendras que usar los specs del mensaje entrante para los mensajes entrantes; al igual que, los specs de los mensajes salientes, para los salientes. Estos dos templates fueron los ultimos que hice; por lo tanto, no estan funcionales.

Ademas, cuando empece a trabajar el navigation(juntar todo), decidi usar ng-view para los templates grandes (dialer, contacts, edit contact, etc) & ng-if para el rightBar. Previamente estaba usando ng-view para el right bar & ng-if para los templates grandes. Ahora mismo el rightbar se compone de dos items (calls y chat) deje chats implementado con ng-if; hay que hacer lo mismo para calls. Al igual que en los templates grandes deje la mayoria programado en ng-view; hacer lo mismo para el screen de llamada y el chat.

La funcion dial,que se encarga de escribir numeros en el inut al oprimirlos, dentro del dialerController, esta comentada debido a que el input de async hace que no puedas escribir numeros en el. Estaba pensando en cambiar el input desde el md al input normal, rapido que el usuario oprima un boton, para que tenga tanto la funcion de buscar contactos async, como la de marcar...

Finalmente al cambiar de ng-if a ng-view y viceversa, hay directives inservibles. Para usar ng-view, no necesitas directives; para ng-if si.

** Faltan detalles como feedbacks y hacerle una prueba detallada a todas las cosas. Ya que no la he hecho.