MY-MEDICAL-ORGANIZER: A MOBILE APPLICATION TO HELP PEOPLE WITH DIABETES

January 25, 2022

THE APP

WHAT IT IS

A mobile app to help Type 1 Diabetics manage medical contact, prescription, and appointment data in once place—with a calculator to facilitate medical supply packing when traveling

WHY IT'S NECESSARY

The organizational demands of living with a chronic illness like diabetes can be overwhelming—the idea of this app is to help relieve some of the pressure

WHO IT'S FOR

People living with diabetes (or anyone with a chronic illness that requires lots of maintenance / management)

WHY I MADE THIS APP

PERSONAL MOTIVATION

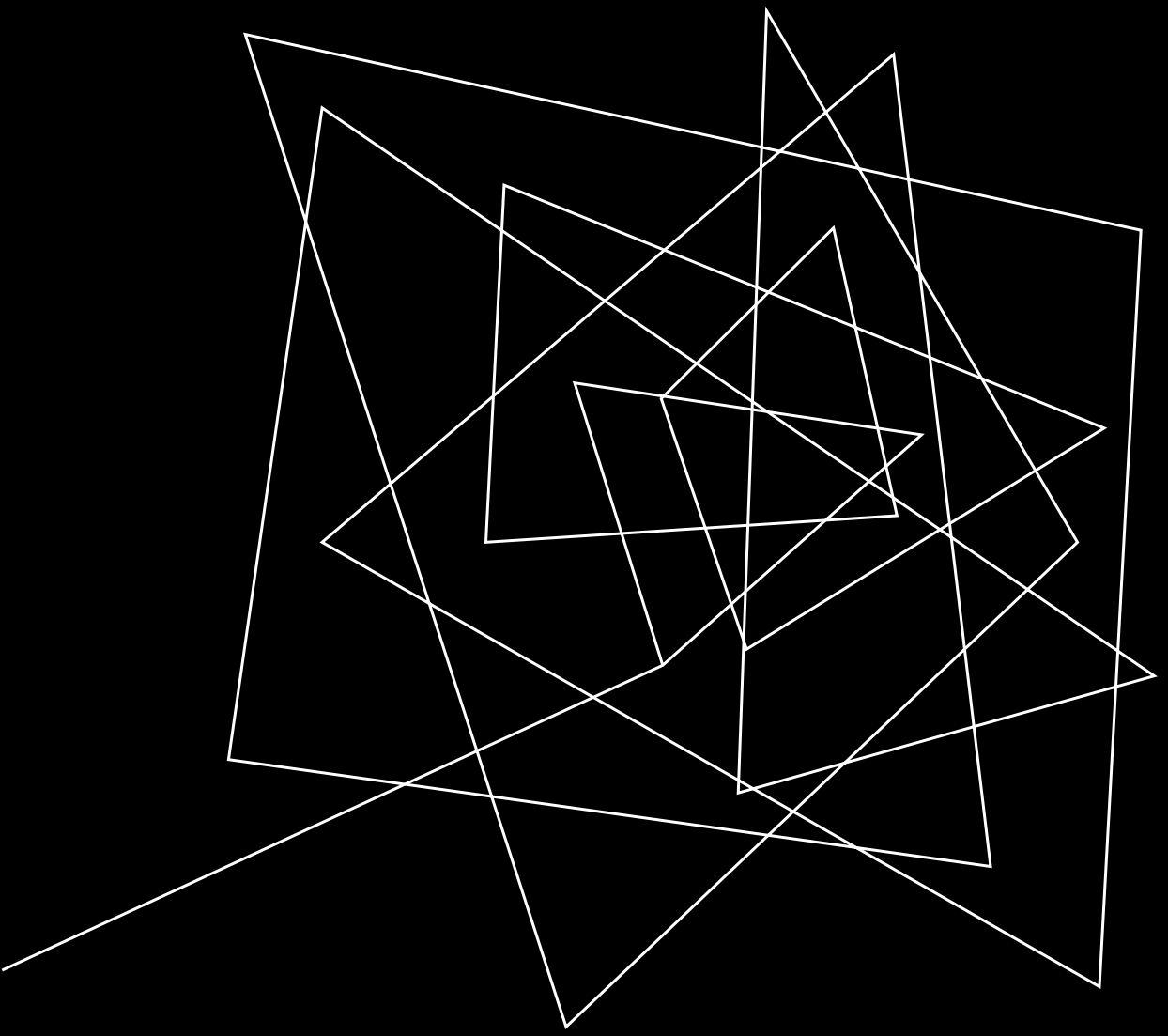
I've lived with Type 1 Diabetes for twenty-three years, and tried many an application to help me manage and keep track of my "diabetes life"—but I found that most were too complicated, buggy, or didn't have the features I actually wanted. I wanted to design an application that was simple, and fulfilled a few basic needs.

ORGANIZATION

Although the main features of this app (contact list, calendar) can easily be used through other apps, my idea in including them in this app was to keep all medical data in one place, reducing search times in the event of needing to access any of this information.

DECISION FATIGUE

The calculator feature of this app relieves the decision / planning fatigue in packing medical supplies while traveling. I've made a million to-do lists for these items over the years, and I thought it was the perfect thing to automate, as it would relieve some of the stress of worrying if I'd forgotten something.



THE PROCESS

1. USER STORIES

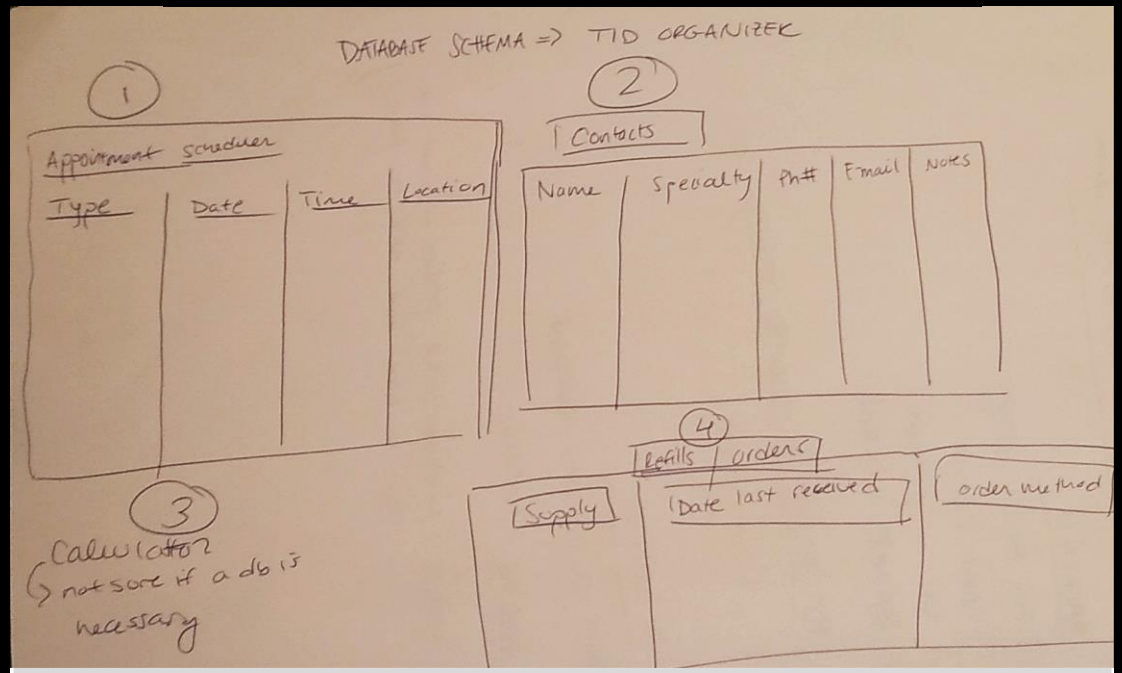
#USER-STORIES: travel calculator

	User Story	Acceptance Criteria
User	*As a* … T1D who likes to travel,	❑ A calculator that **accepts** input (type of travel—ie. air, car, train, etc., length of trip, and supplies used per day) and **returns**: checklist of necessary items (interactive)
Need	*I want* … to have a calculator that will me which / how many supplies I need to bring with me, and will generate a reactive check-list,	
Reason	*So that* … I can focus on other things in travel prep and feel confident that I am not forgetting anything.	

#USER-STORIES: prescription / ordering reminder calendar

	User Story	Acceptance Criteria
User	*As a* … T1D, again, with a busy schedule,	❑ A supply function that takes input (when supplies were received or prescription filled, and when they will need to be ordered again → adds to calendar and sends reminder when date is approaching + contact info for ordering
Need	*I want* … a tool to facilitate ordering supplies / filling prescriptions,	
Reason	*So that* … living with diabetes feels less annoying	

2. DATABASE DESIGN



#DATABASE-SCHEMA

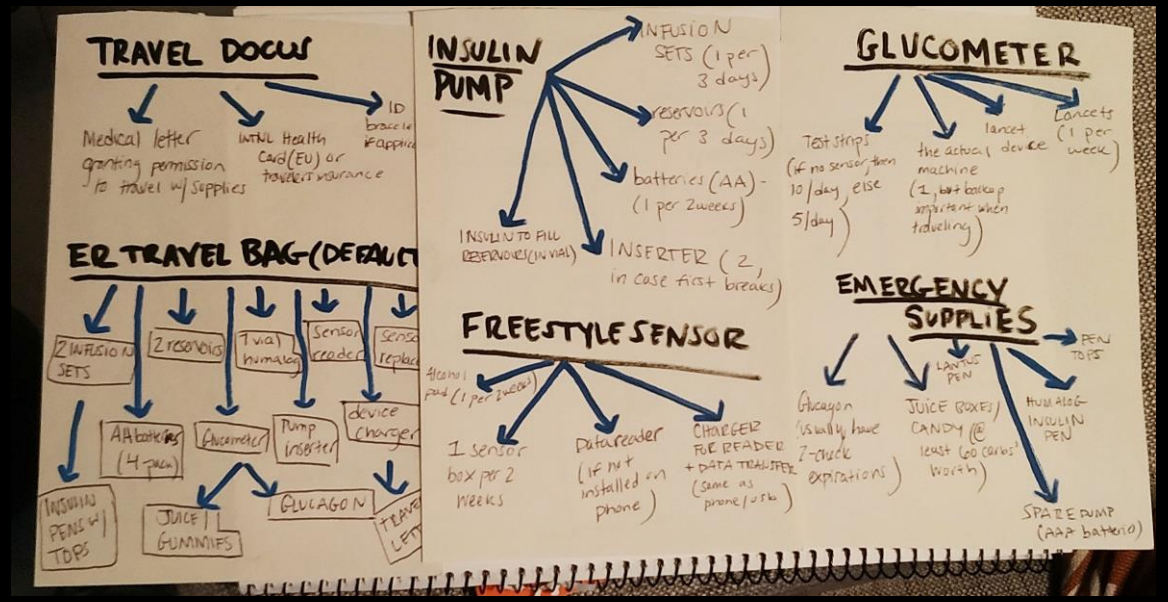
Contacts
Id
Firstname
Lastname
Speciality
phone number
e-mail
Office name
Notes

Appointments
id
type
date
time
location
labwork
notes

Supplies / Meds
id
med-name
date received
duration
order-method

Ideally, I'd like to input order type & contact (some things have to be ordered at pharmacy in person, some automatic, some have to call health center to request)

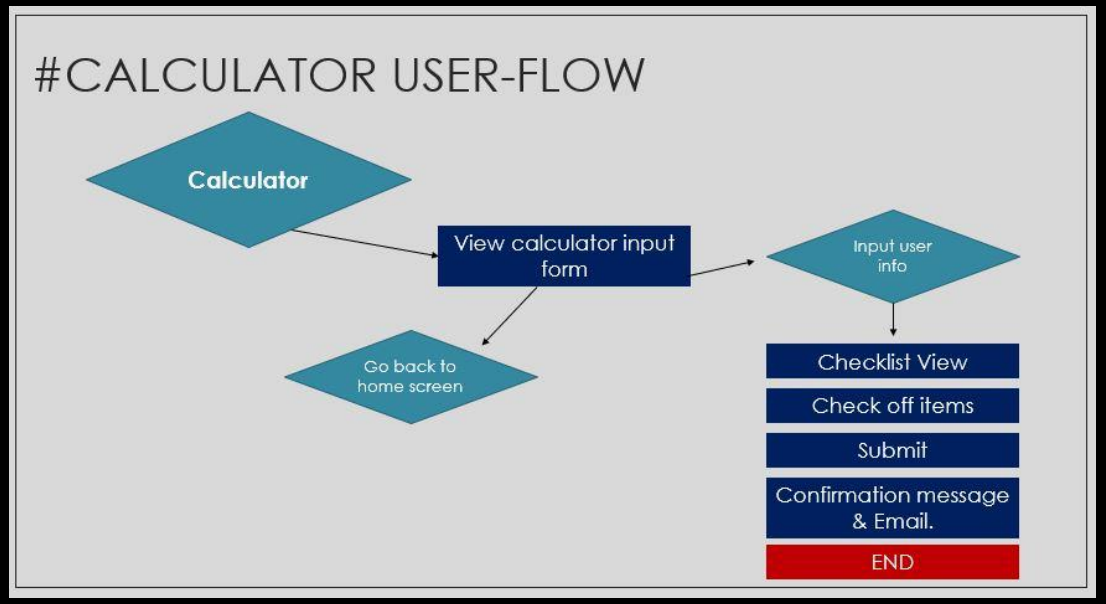
3. CALCULATOR DESIGN

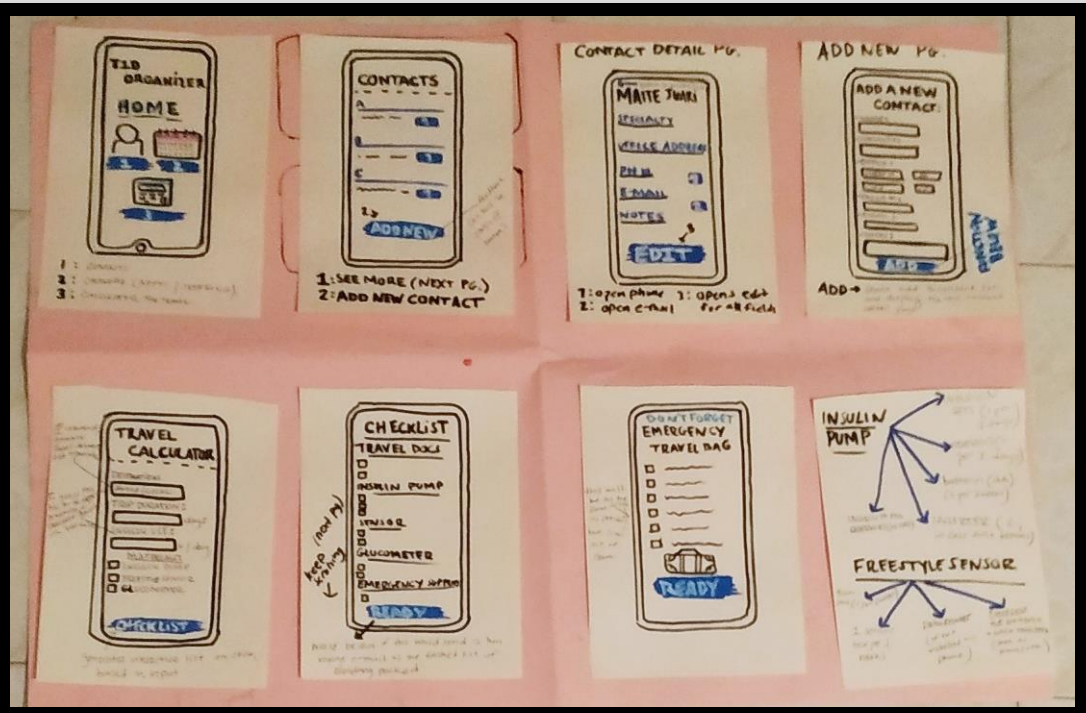


TRAVEL BAG (DEFAULT)

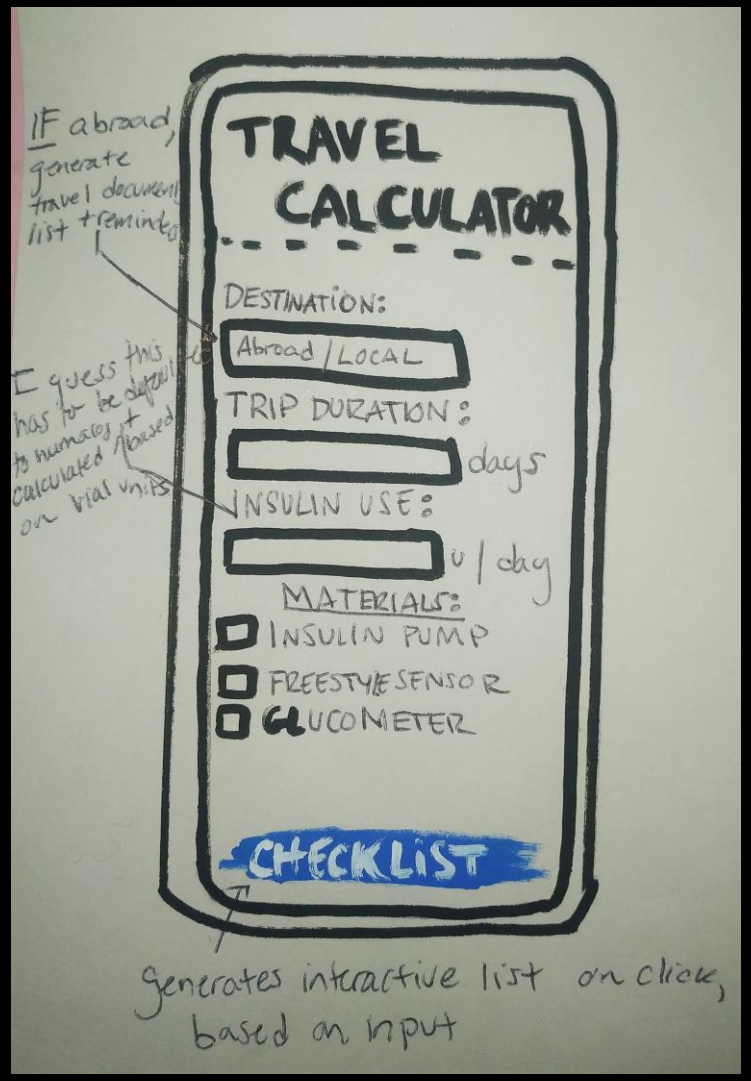
Material	Amount	Code Equation
Glucometer	1	These are all of the supplies that I bring with me in a smaller bag (so it can be accessed more easily if anything is needed from it) on longer trips –so the amounts are default and will be the same no matter how long the trip is. So I believe this would render by default with the same amounts each time in a portion of the checklist titled "Emergency Travel Bag".
Infusion sets	2	
Reservoirs	2	
Humalog (vial)	1	
Pump Inserter	1	
Sensor Reader	1	
Sensor replacement	1	
AA Batteries	1 pack of 4	
Lantus Pen	1	
Humalog Pen	1	
Insuiln Pen tops	10	
Device Charger	1	
Travel Letter	1	
Glucagon	1	
Juice / Candy	At least 60 carbohydrates' worth	

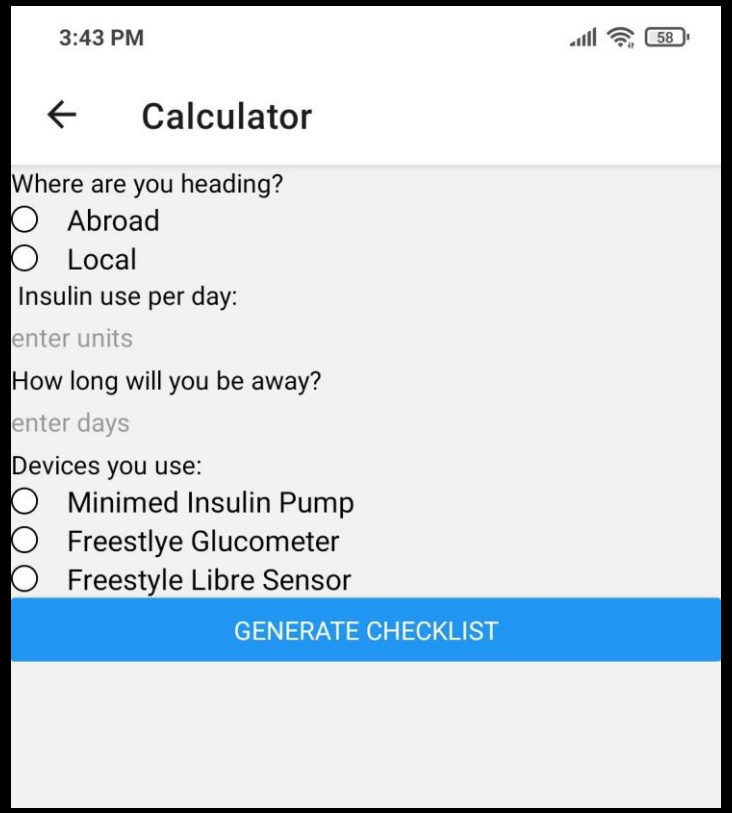
4. SCREEN SKETCHES & USER FLOW



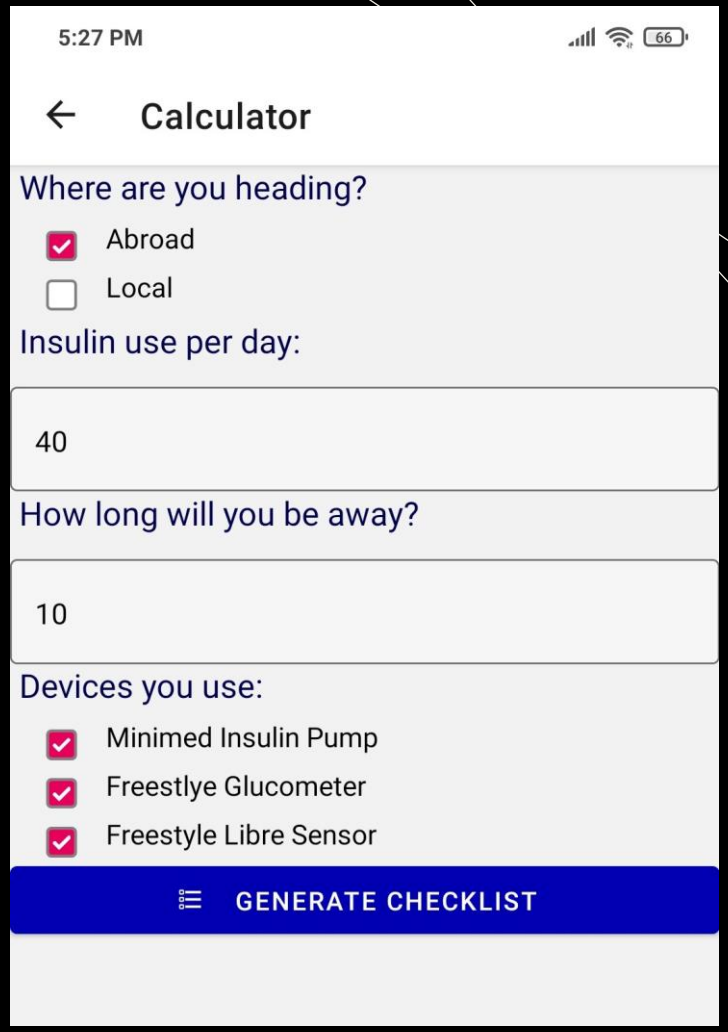


FRONT-END DEVELOPMENT



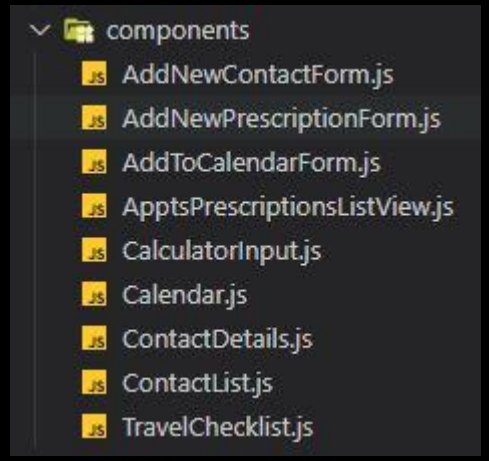


Progression of the calculator component.



FRONT-END DEVELOPMENT: THE CODE SIDE

```
--Declare data you'd like to pass before app return area
--Wrap the return Stack Navigator in the context you created
--Add values to the Context.Provider-- the values declared here will be available throughout the app
--They can be data, functions, etc.
*/
export const Context1 = React.createContext(null);

function App() {
  //I want to be able to access the data obtained for appts., materials,
  //and contacts throughout the app, so I declare them all here
  const [contacts, setContacts] = useState([]);
  const [appointments, setAppointments] = useState([]);
  const [materials, setMaterials] = useState([]);

  //This grabs all the contacts from database and sets state
  useEffect(() => {
    fetch("http://localhost:5000/contacts")
      .then((res) => {
```



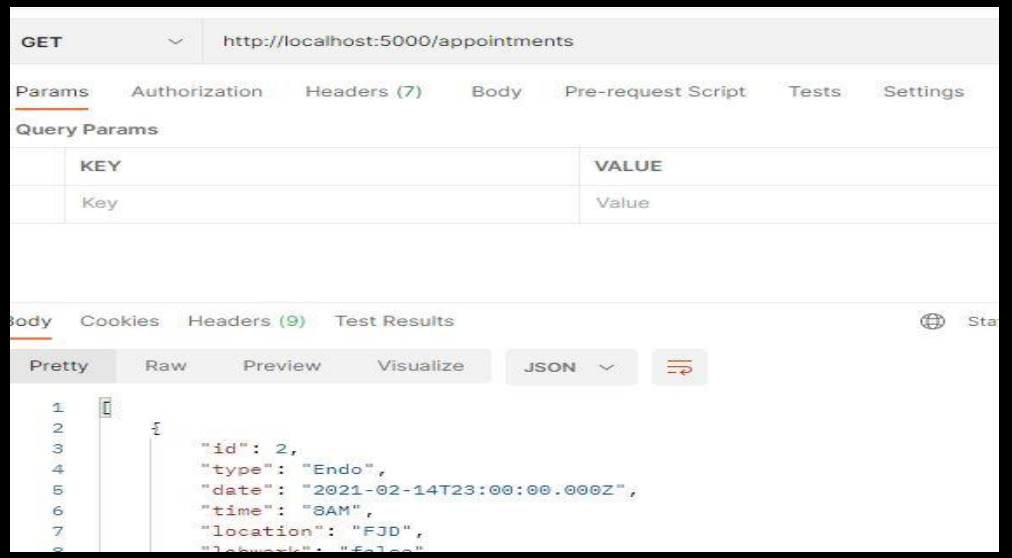

```
function ContactDetails({ route, navigation }) {
  const id = route.params.id;
  const [contact, setContact] = useState([]);

  const context = useContext(Context1);

  const styles = StyleSheet.create({
    title: {
      fontWeight: "normal",
      color: "#00004d",
      textDecorationLine: "underline",
    },
    name: {
      fontSize: 40,
      lineHeight: 40,
      fontFamily: "sans-serif",
      fontWeight: "normal",
      paddingBottom: 20,
    },
    row: {
      flex: 1,
      paddingVertical: 25,
      paddingHorizontal: 15,
      flexDirection: "row",
      justifyContent: "space-between",
      borderBottomWidth: 1,
      borderBottomColor: "white",
```

BACK-END DEVELOPMENT

#API-ROUTES-PLAN

URI	HTTP Method	Description	Request Object	Response Object
/contacts	GET	Retreives the list of contacts objects		results(contacts).data
/appointments	GET	Retrieves the list of appointments		results(appointments).data
/addcontact	POST	Adds a new contact to the list	Object with new contact data	Sends contact list back
/contacts/:id	DELETE	Deletes a contact from the list	Contact ID parameter	Sends back updated contact list

```
let sql =
  "DROP TABLE if exists contacts; CREATE TABLE contacts(id INT
let sql2 =
  "DROP TABLE if exists appointments; CREATE TABLE appointments
let sql3 =
  "DROP TABLE if exists materials; CREATE TABLE materials (id I
```




```
//This adds a new contact to the list
app.post("/addcontact", (req, res) => {
  db(
    `INSERT INTO contacts (firstname, lastname, specialty, phonenumber, email, officename, notes) VALUES ("${re
  )
    .then((result) =>
      db("SELECT * FROM contacts;") //this is the MySql query
        .then((results) => {
          res.send(results.data);
        })
    )
    .catch((err) => res.status(500).send(err));
});

//This gets a contact with a given id
app.get("/contacts/:id", (req, res) => {
  db(`select * from contacts where id=("${req.params.id}");`)
    .then((results) => res.send(results.data))
    .catch((err) => res.status(500).send(err));
```

Pictured here:

- API planning
- Table creation
- Route writing & testing

TECHNOLOGIES AND RESOURCES

Front-end

-React Native with Expo
-React Navigation
-React Native Paper

Back-end

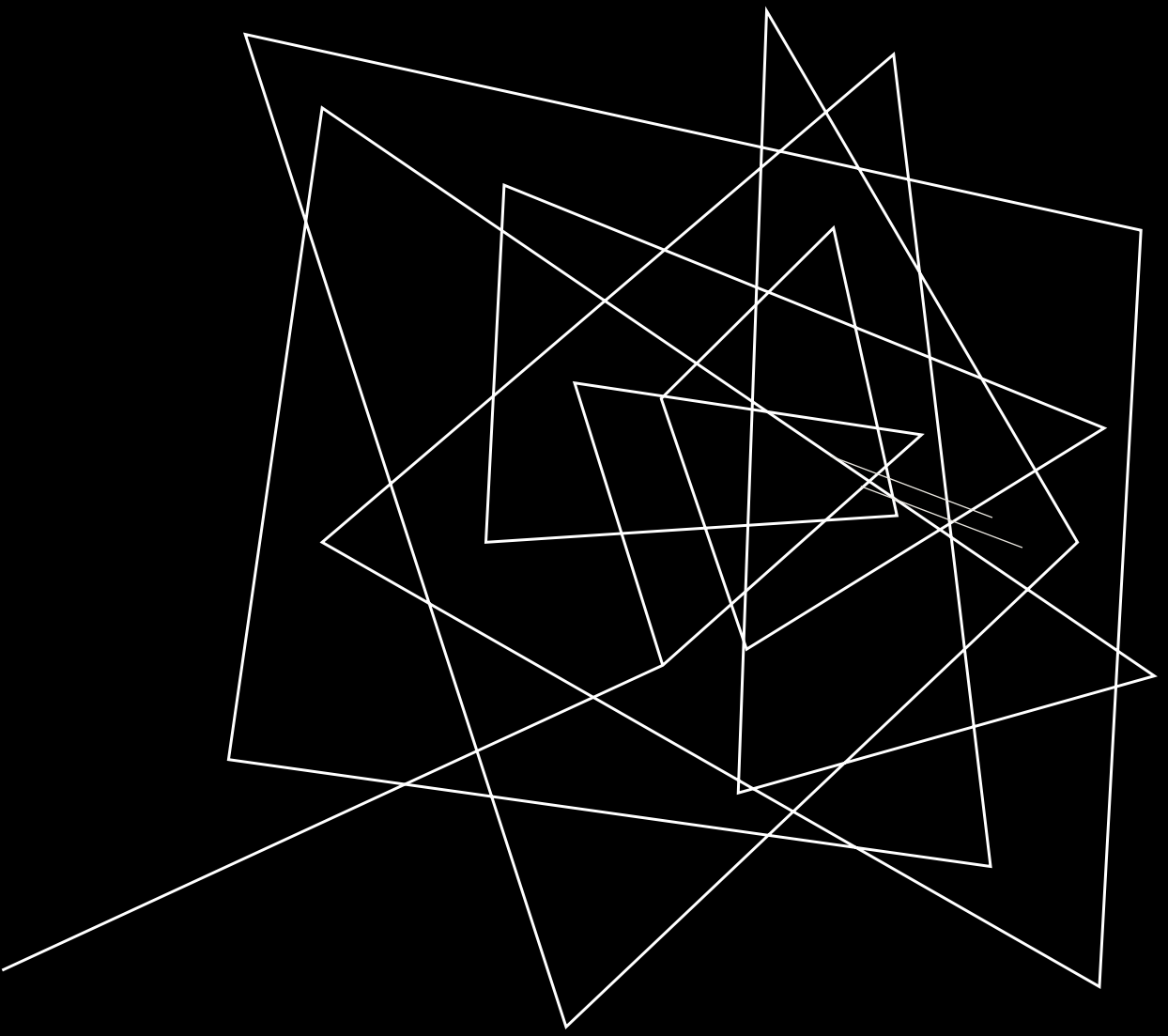
-MySQL: Database
-Express JS: Server
-Postman for Route Testing

Misc.

-React Native Calendars
-Material Design Icons

Resources

- Google, all the time
-Stack Overflow
-React Native Documentation
-React Navigation documentation
-Dev.to



LOOKING BACK (AND FORWARD!)

REFLECTIONS

Most Difficult

- Learning React Native on the fly
- React Navigation
- Setup errors
- useContext
- Styling

Lessons Learned

- React native, passing parameters, using context throughout an app

Proudest Moments

- Using context with fetch to update data immediately on-screen
- Finally understanding the stack navigator and how to use it
- The moment my app actually displayed on the phone screen (I couldn't program anything four months ago and now I have a semi-functional app!)

FUTURE FEATURES

CURRENT FEATURE	FUTURE IMPROVEMENTS
Delete appointments and prescription button on upcoming appts/prescription view (has a bug!)	• Bug is explained in code
Calculator that takes input and generates an interactive checklist	• The ability to save the checklist to work on later • Animation on submit and list of packed items e-mailed to the user
Contact List that has add, delete, and view details functionality	• Add an edit function to contact details • Display contacts by specialty and in alphabetical order • Add “call now” or “email” buttons to contacts with this data
Appointment and Prescription List View that displays items in ascending order by date	• Ability to show only appts / prescription renewals coming up over the next month • Get history of all appointments / prescriptions function
Calendar (currently implemented but non-functional)	• Connect with backend / add appointment and prescription forms so that these events are displayed visually on calendar • Set up push notifications for upcoming appointments and renewals