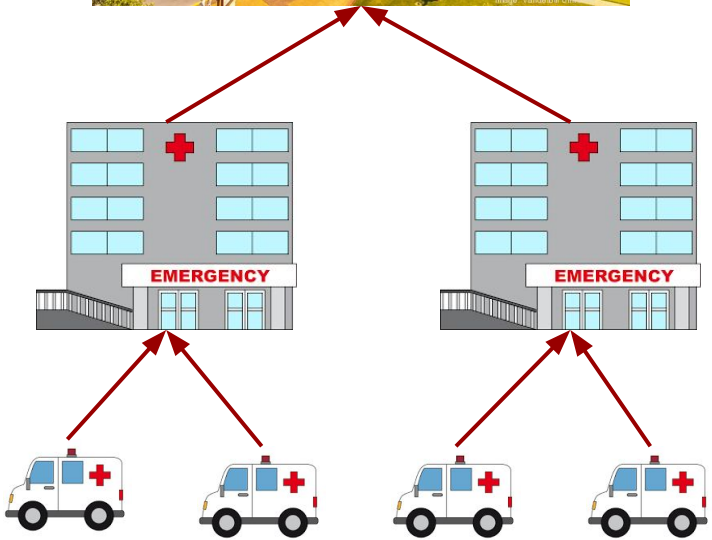
Artificial Intelligence Based Burn Resuscitation

BMExtra Group:
Jacob Ayers (BME), Hannah Kang (BME), Dominique Szymkiewicz (BME), Nora Ward (BME), Thomas Yates (BME), Eric Yeats (CompE)

Contact:
Avinash Kumar M.D.

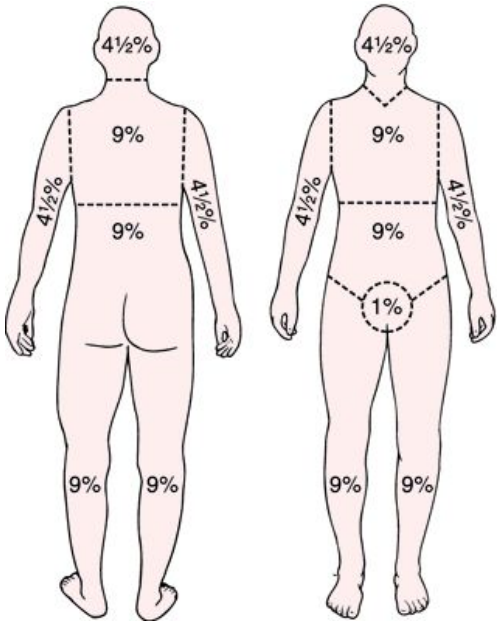
VUMC Burn ICU

- Vanderbilt Burn ICU
  - Level 1 Burn Center
  - 630 new admissions per year
    - Majority transferred from E.R.
  - Primary Contact: Dr Avinash Kumar



Problem Statement

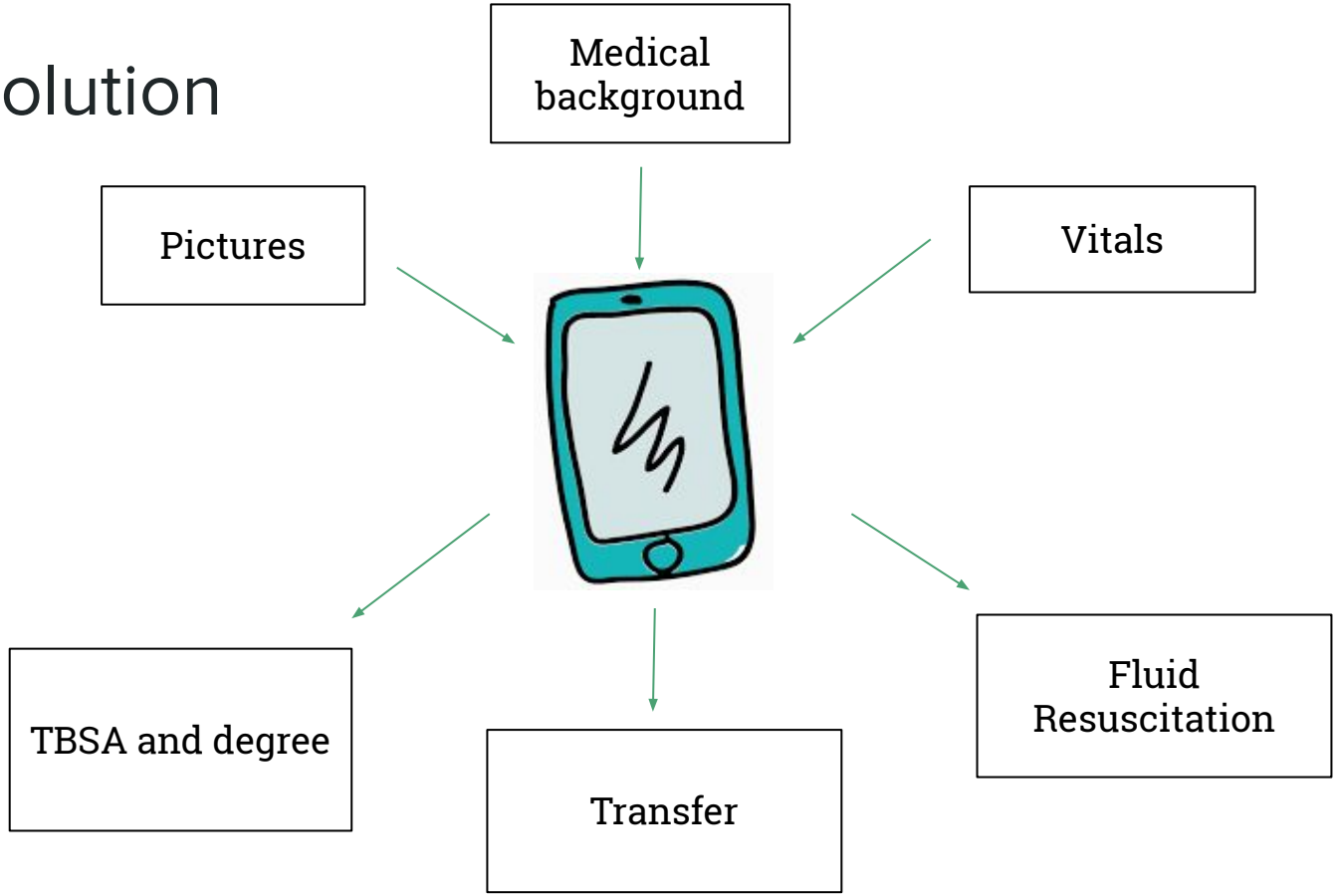
- Current System:
  - Wallace Rules of Nine (Adult)
  - Rules of Eight/Palm method
- Problems:
  - Overestimation of burn percentage
    - 79% of TBSA was estimated inaccurately
      - ½ of these burns overestimated by ≥ 5%
  - Overburden Burn centers with patients
  - Does not account for different body types
- **Goal: Develop system to rapidly and accurately determine TBSA**



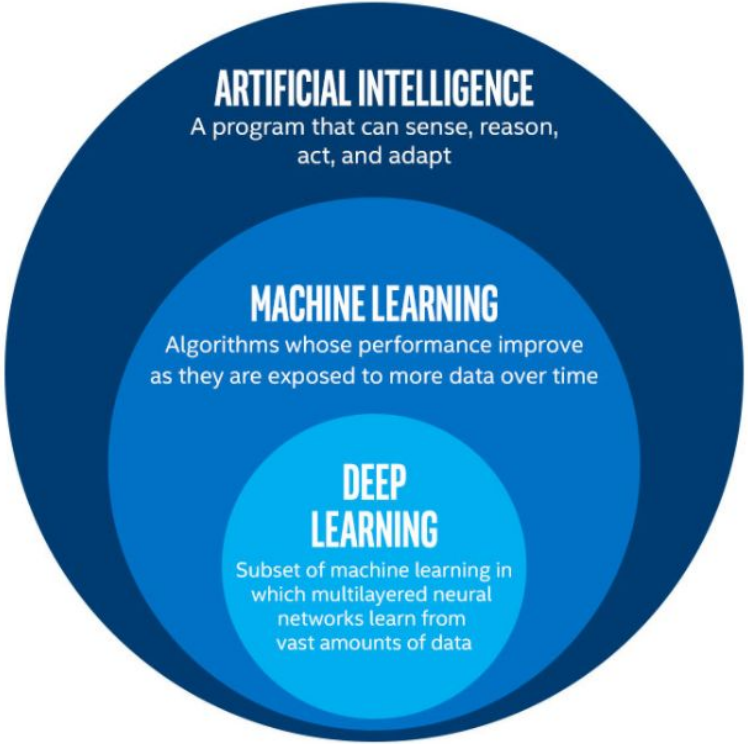
Needs Assessment

1. Infrastructure Compatibility
2. Safety
3. Patient Efficacy
4. Performance Capabilities
5. Cost Efficacy

Our Solution

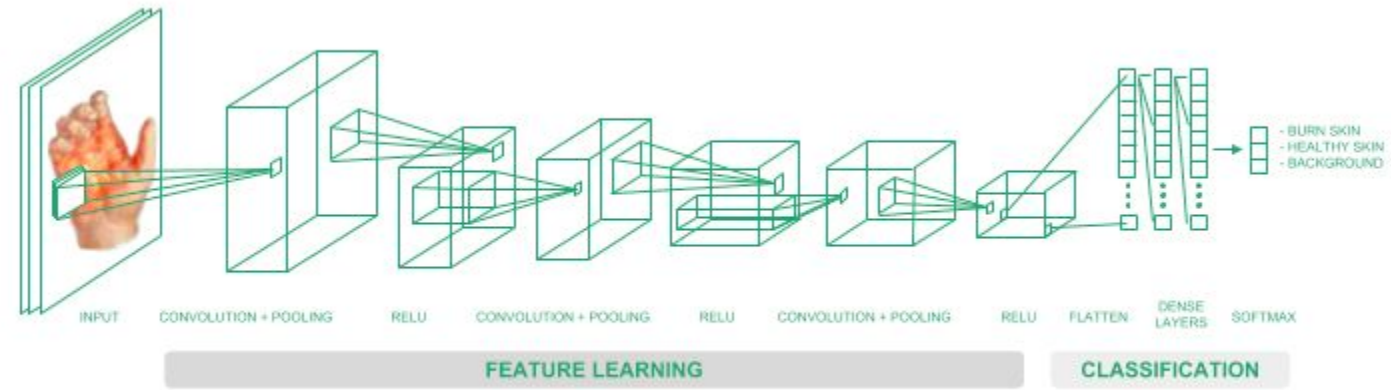


Our Approach

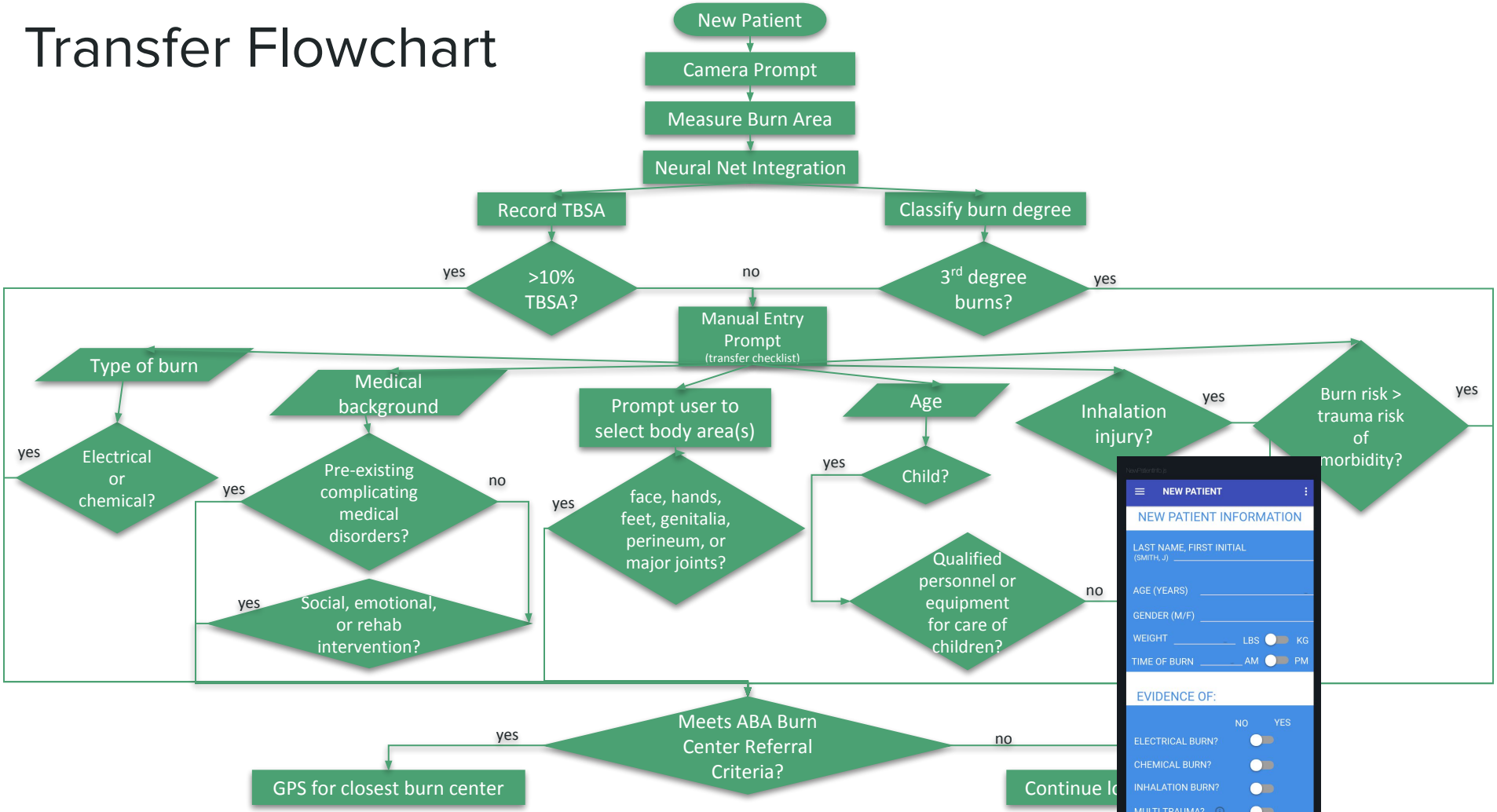


Neural Network Model

- Convolutional Neural Network
  - 50x50x3 Images ➜ Softmax Output



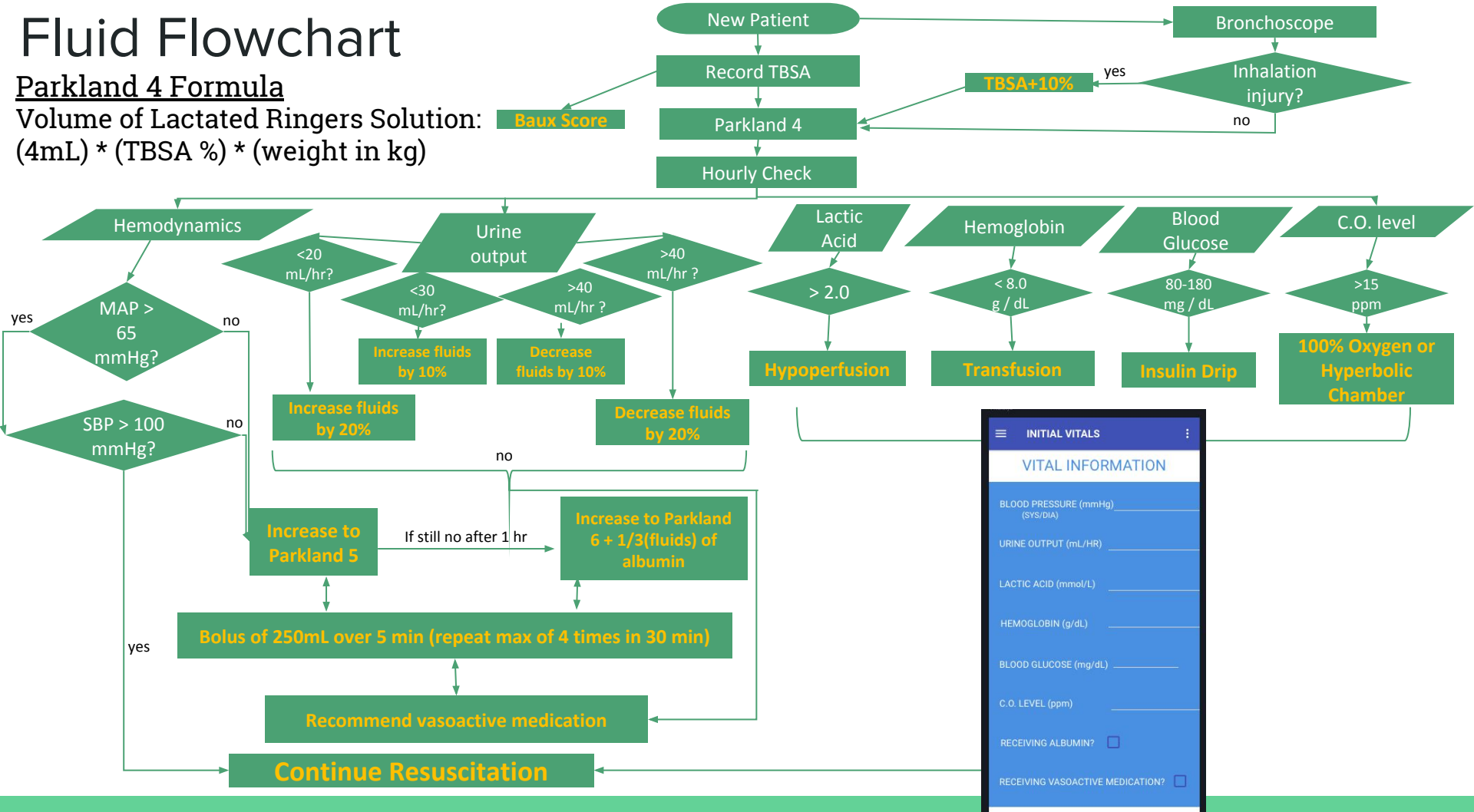
Transfer Flowchart



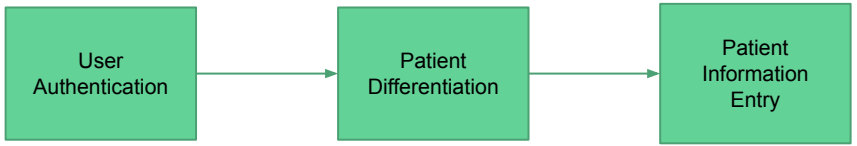
Fluid Flowchart

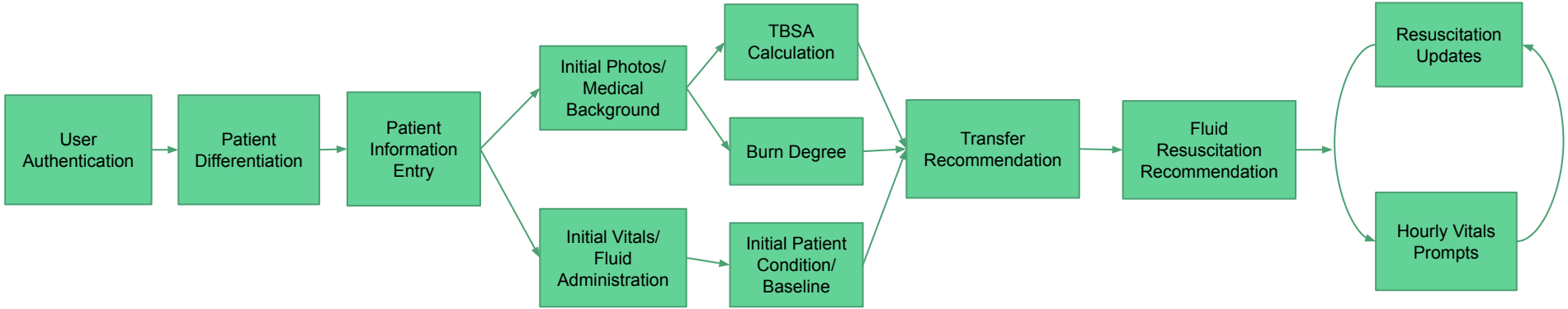
Parkland 4 Formula

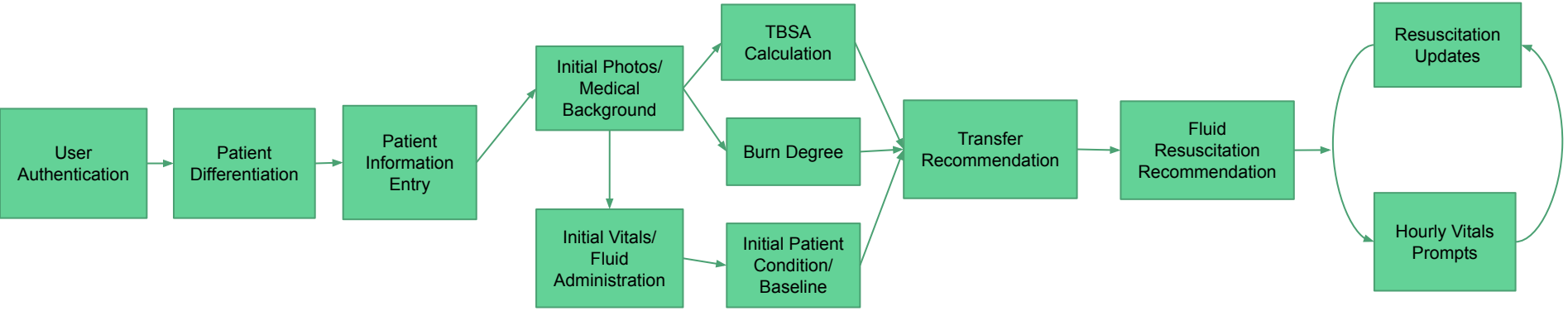
Volume of Lactated Ringers Solution:
(4mL) * (TBSA %) * (weight in kg)

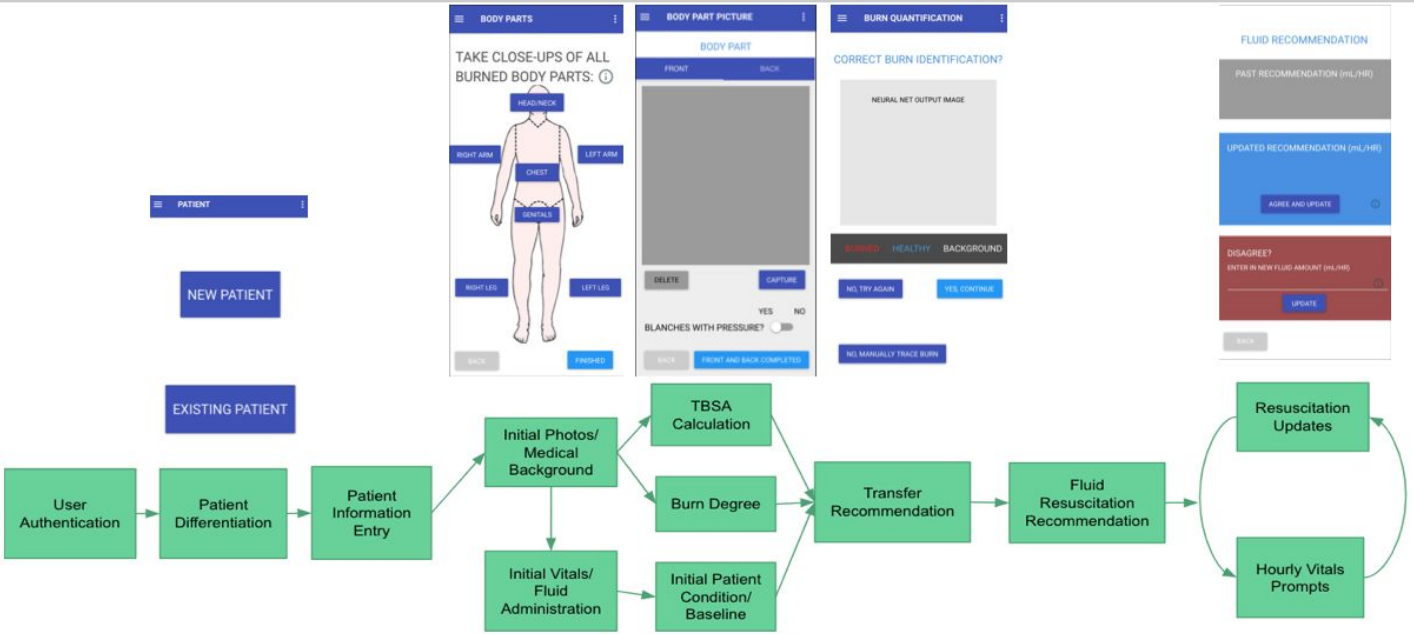


User Interface - Basic Design Ideas




User Authentication
Patient Differentiation
Patient Information Entry
Initial Photos/ Medical Background
Initial Vitals/ Fluid Administration
TBSA Calculation
Burn Degree
Initial Patient Condition/ Baseline
Transfer Recommendation
Fluid Resuscitation Recommendation
Resuscitation Updates
Hourly Vitals Prompts


User Authentication
Patient Differentiation
Patient Information Entry
Initial Photos/ Medical Background
TBSA Calculation
Burn Degree
Initial Vitals/ Fluid Administration
Initial Patient Condition/ Baseline
Transfer Recommendation
Fluid Resuscitation Recommendation
Resuscitation Updates
Hourly Vitals Prompts


PATIENT
NEW PATIENT
EXISTING PATIENT
BODY PARTS
TAKE CLOSE-UPS OF ALL BURNED BODY PARTS:
HEAD/NECK
RIGHT ARM
LEFT ARM
CHEST
GENITALS
RIGHT LEG
LEFT LEG
BACK
FINISHED
BODY PART PICTURE
BODY PART
FRONT
BACK
DELETE
CAPTURE
YES
NO
BLANCHES WITH PRESSURE?
BACK
FRONT AND BACK COMPLETED
BURN QUANTIFICATION
CORRECT BURN IDENTIFICATION?
NEURAL NET OUTPUT IMAGE
BURNED
HEALTHY
BACKGROUND
NO, TRY AGAIN
YES, CONTINUE
NO, MANUALLY TRACE BURN
FLUID RECOMMENDATION
PAST RECOMMENDATION (mL/HR)
UPDATED RECOMMENDATION (mL/HR)
AGREE AND UPDATE
DISAGREE?
ENTER IN NEW FLUID AMOUNT (mL/HR)
UPDATE
BACK
User Authentication
Patient Differentiation
Patient Information Entry
Initial Photos/ Medical Background
Initial Vitals/ Fluid Administration
TBSA Calculation
Burn Degree
Initial Patient Condition/ Baseline
Transfer Recommendation
Fluid Resuscitation Recommendation
Resuscitation Updates
Hourly Vitals Prompts
Burn Resuscitation Network
USERNAME
PASSWORD
LOG IN
Create an Account
NEW PATIENT
NEW PATIENT INFORMATION
LAST NAME, FIRST INITIAL
(SMITH, J)
AGE (YEARS)
GENDER (M/F)
WEIGHT
LBS
KG
TIME OF BURN
AM
PM
EVIDENCE OF:
NO
YES
ELECTRICAL BURN?
CHEMICAL BURN?
INHALATION BURN?
MULTI TRAUMA?
BACK
CONTINUE
INITIAL VITALS
VITAL INFORMAITON
BLOOD PRESSURE (mmHg)
(SYS/DIA)
URINE OUTPUT (mL/HR)
LACTIC ACID (mmol/L)
HEMOGLOBIN (g/dL)
BLOOD GLUCOSE (mg/dL)
C.O. LEVEL (ppm)
RECEIVING ALBUMIN?
RECEIVING VASOACTIVE MEDICATION?
BACK
CONTINUE TO CHART
INITIAL INPUTS
FLUID INFORMATION
FLUID DELIVERED SINCE ADMISSION (ML)
SOLUTION
TOTAL BURN SURFACE AREA (TBSA)
BURN DEGREE
FLUID RESUCITATION RECOMMENDATION
TRANSFER RECOMMENDED
BACK
ADD VITALS
EXISTING PATIENT
EXISTING PATIENT
LAST NAME, FIRST INITIAL
(SMITH, J)
BACK
CONTINUE TO CHART

Current Progress of App

- Features
  - Persistent Data storage
  - ConvNet and TBSA calculation algorithm onboard
  - Burn classification overlay
  - Patient Information and Vitals Input
- Future Features
  - Treatment recommendation
  - Vitals and recommended Treatment chart visualization

Validation Metrics

- Compare our app to current gold standard
  - Similarity Metric
    - Compare the “masks” created by code
  - Bland-Altman Comparison
- Determine effect on patient care
  - Average time
  - Compare fluid recommendations
  - Quality of Care

Capturing Validation Metrics

- Excel spreadsheet
  - Widely used within the Burn ICU to calculate TBSA
  - Easily formatted to record/analyze data
- Provider Survey
  - Feedback on integration/ease of use
  - Look for potential errors
  - Qualitatively determine effect of this app

Next Steps

- Finalize Application Deployment
- Validation/Testing
  - Creation of capture forms
  - Discussion on validation timeline
- Poster Completion

TBSA Algorithm and ConvNet on App

```java
@ReactMethod
public void calcTBSA(String imgb64, Callback errorCallback, Callback successCallback) {
    try {
        // convert Base64 to single-precision floating point
        BitmapFactory.Options options = new BitmapFactory.Options();
        options.inDither = true;
        options.inPreferredConfig = Bitmap.Config.ARGB_8888;

        byte[] decodedString = Base64.decode(imgb64, Base64.DEFAULT);
        Bitmap image = BitmapFactory.decodeByteArray(decodedString, offset: 0, decodedString.length);

        int[] results = {0, 0, 0};
        // cut up image into 50x50 images, should already be BGR
        // feed each image to the neural network
        for (int row = 0; row < image.getHeight() - INPUT_SIZE; row+=INPUT_SIZE) {
            for (int col = 0; col < image.getWidth() - INPUT_SIZE; col += INPUT_SIZE) {
                // slice image and get bytebuffer
                Bitmap subimage = Bitmap.createBitmap(image, col, row, INPUT_SIZE, INPUT_SIZE);
                ByteBuffer byteBuffer = convertBitmapToByteBuffer(subimage);
                float[][] result = new float[1][labels.size()];
                tflite.run(byteBuffer, result);
                int ind = argMax(result);
                results[ind] += 1;
            }
        }
        successCallback.invoke( ...args: Double.toString( d: (double)results[0]/(results[0]+results[1])));
    } catch (Exception e) {
        errorCallback.invoke(e.getMessage());
    }
}
```

```
import React, { Component } from "react";
import Button117 from "../symbols/button117";
import { Center } from "@builderx/utils";
import Button612 from "../symbols/button612";
import { View, StyleSheet, Text } from "react-native";

export default class EndRes extends Component {
  render() {
    return (
      <View style={styles.root}>
        <Center vertical>
          <View style={styles.rect} />
        </Center>
        <Center horizontal>
          <Button117 style={styles.button117} />
```

```
export default class SecurityAuth extends Component {
  render() {
    return (
      <View style={styles.root}>
        <Image
          source={require("../assets/a3d677392ee343199bc8e0bfbba7037f_(1).jpeg")}
          style={styles.logo}
        />
        <View style={{...styles.gray, flex: 0.6, flexDirection: 'column', justifyContent: 'flex-start', alignItems: 'stretch'}} >
          <View style={{flex: 1, flexDirection: 'row', alignItems: 'center'}}>
            <Text style={styles.text}>USERNAME</Text>
            <DisabledTextbox style={styles.DisabledTextbox} />
          </View>
          <View style={{flex: 1, flexDirection: 'row', alignItems: 'center'}}>
            <Text style={styles.text}>PASSWORD</Text>
            <DisabledTextbox style={styles.DisabledTextbox} />
          </View>
          <View style={{flex: 1, flexDirection: 'row', justifyContent: 'center', alignItems: 'center'}}>
            <Button101
              style={{...styles.button}}
              root={() => {
                this.props.navigation.push("Patient");
              }}
              onPress={() => {
                this.props.navigation.push("Patient");
              }}
            />
          </View>
        </View>
```