Lab 3

VITAL SIGNS – UPDATE & ABNORMAL

Variable Scope

- **Class Level Scope**

```
public class User {

    private String username;

}
```

- Method Level Scope

```
public static void main(String[] args) {
    int x = 5;
}
```

- **Loop Scope**

```
public static void main(String[] args) {
    for (int x = 0; x < 5; x++) {
        System.out.println("Loop " + x);
    }
}
```

ArrayList Container

- Implements the List interface
- Can grow and shrink as needed
- Ordered collection (i.e. sequence)
- Supports random access

- Online Information

https://docs.oracle.com/javase/tutorial/collections/

ArrayList Container

Instantiation:

List<T> list = new ArrayList>();

Method:

list.add(val);// add value

list.get(index);// index: 0 – (length -1)

list.size();//

list.remove(index);//index or object

Traverse:

for(T t: lsit) {t.action()}

For(int i=0;i<list.size();i++) {list.get(i).action()}

Passing by value/reference

```
// Pass by value
System.out.println("Passing value to the method");
test.testMethod(0);
test.testMethod(5);
test.testMethod(100);
System.out.println("");
```

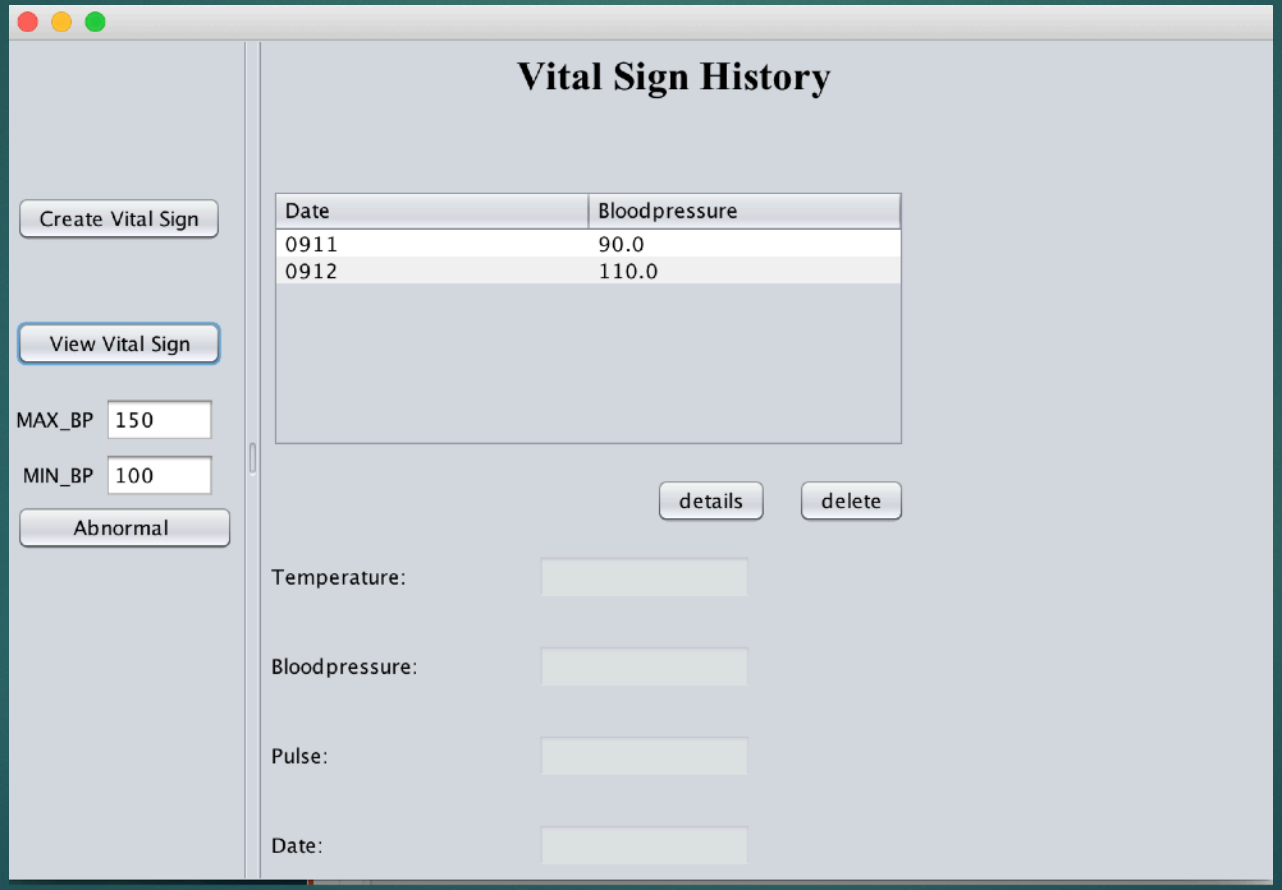
```
class Test {
    public void testMethod(int n) {
        System.out.println("Value of n is: " + n);
    }
}
```

```
// Pass by reference
System.out.println("Passing reference to the method");
Test2 test2 = new Test2();
Value v = new Value(1);
  v.setM(1);
test2.test2Method(v);
v = new Value(9);
test2.test2Method(v);
v = new Value(101);
test2.test2Method(v);
```

```
class Value {
    int m;

    Value(int m){
        this.m = m;
    }

    public int getM() {
        return m;
    }

    public void setM(int m) {
        this.m = m;
    }
}
```

```
class Test2 {
    public void test2Method(Value v){
        System.out.println("Value of m is: " + v.getM());
    }
}
```

Lab2.5 Vital Sign History



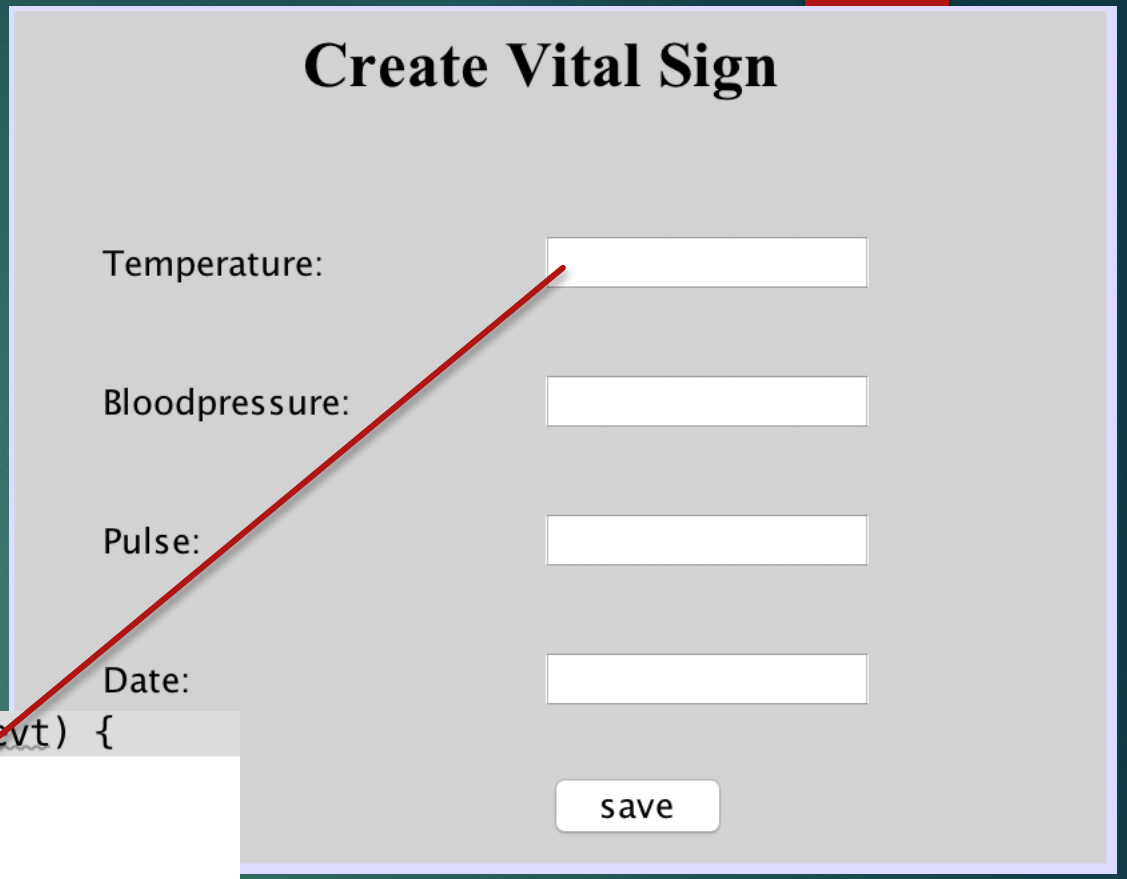
Implement relationship

```java
public class VitalSign {
    private double temperature;
    private double bloodPressure;
    private int pulse;
    private String date;
```

```java
public class VitalSignHistory {
    private List<VitalSign> vitalSignList;

    public VitalSignHistory() {
        this.vitalSignList = new ArrayList<VitalSign>();
    }
    public VitalSign addVitalSign(){
        VitalSign vitalSign = new VitalSign();
        this.vitalSignList.add(vitalSign);
        return vitalSign;
    }
    public void delVitalSign(VitalSign vitalSign){
        this.vitalSignList.remove(vitalSign);
    }

    public List<VitalSign> getVitalSignList (){
        return vitalSignList;
    }
```

```java
private VitalSignHistory vsh;
/**
 * Creates new form MainFrame
 */
public MainFrame() {
    initComponents();
    this.vsh = new VitalSignHistory();
}
```

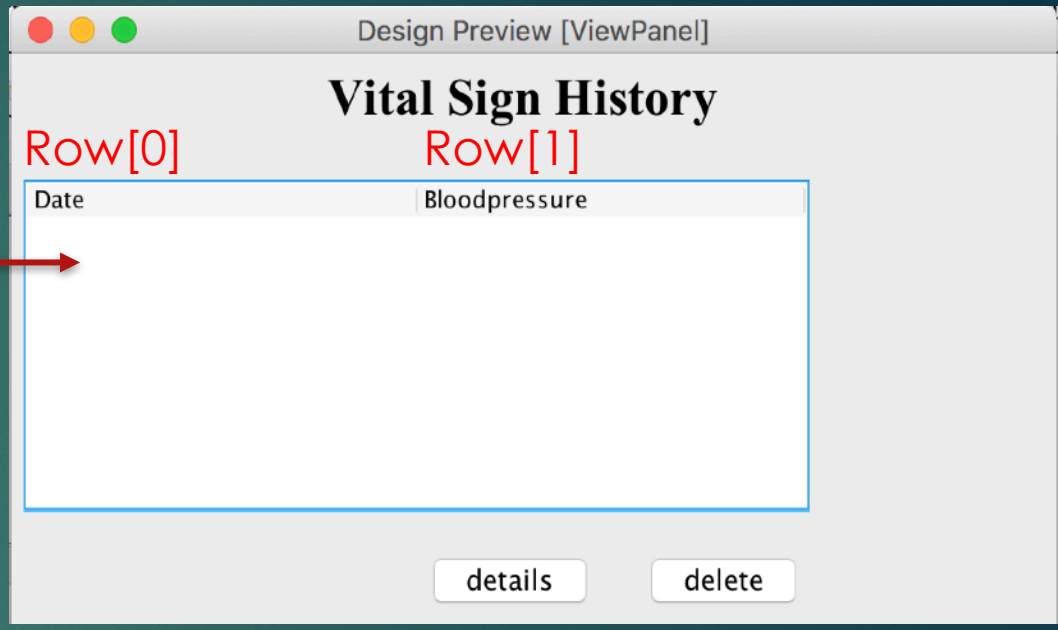
Create Vital Sign




```
private void saveBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    double tmp = Double.parseDouble(tmpTextField.getText());
    double bp = Double.parseDouble(bpTextField.getText());
    int pulse = Integer.parseInt(pulseTextField.getText());
    String date = dateTextField.getText();
    VitalSign vs = vsh.addVitalSign();
    vs.setTemperature(tmp);
    vs.setBloodPressure(bp);
    vs.setPulse(pulse);
    vs.setDate(date);
    JOptionPane.showMessageDialog(null,"Vital Sign Created Successfully");
    resetTxtField();
```

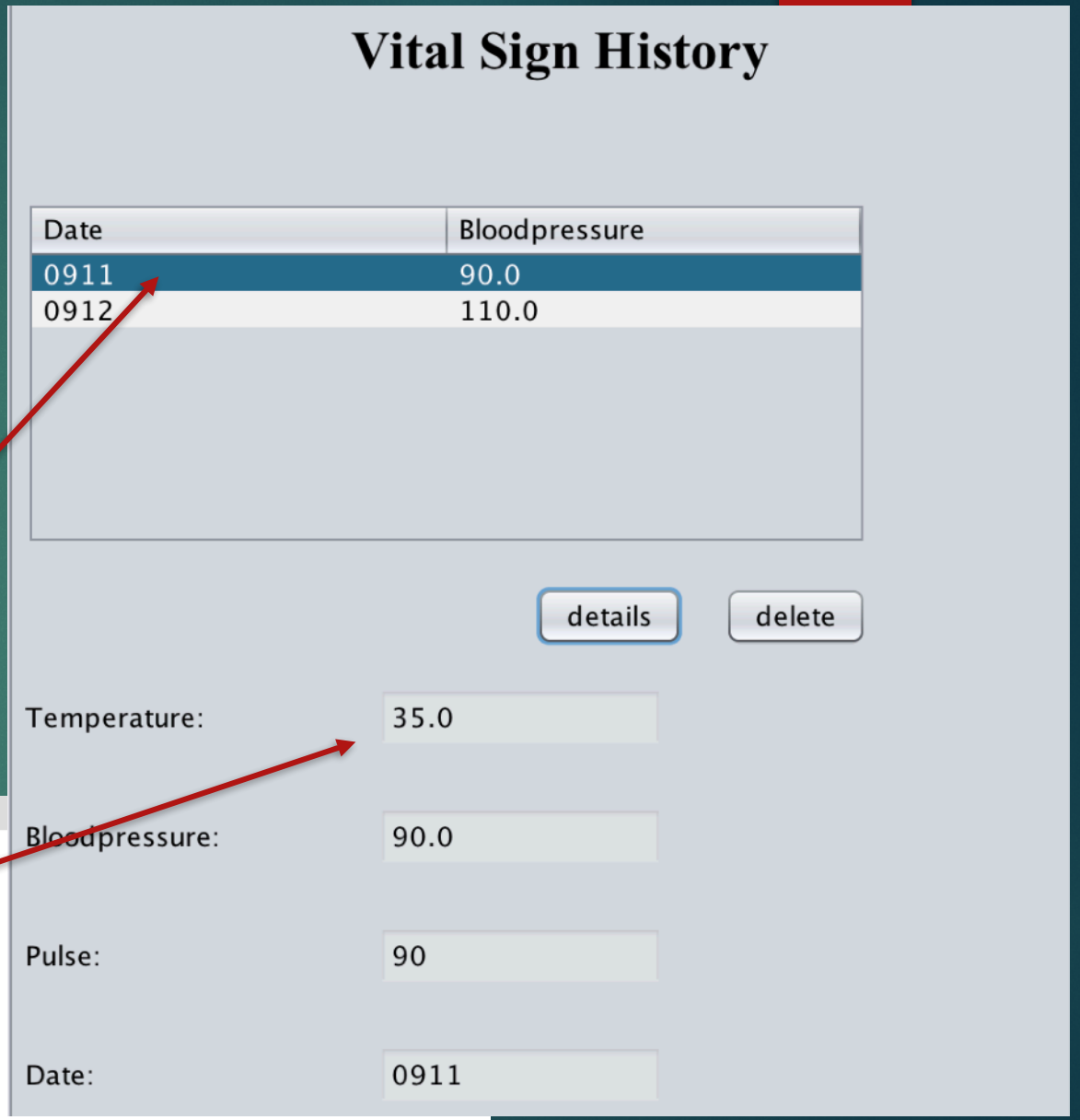
Show vital sign history in JTable

```
private VitalSignHistory vsh;
public ViewPanel(VitalSignHistory vsh) {
    initComponents();
    this.vsh = vsh;
    populateTable();
}
```




```
private void populateTable(){
    DefaultTableModel dtm = (DefaultTableModel) vitalSignTab.getModel();
    dtm.setRowCount(0);
    for (VitalSign vs : vsh.getVitalSignList()){
        Object row[] = new Object [2];
        row[0] = vs;
        row[1] = vs.getBloodPressure();
        dtm.addRow(row);
    }
```

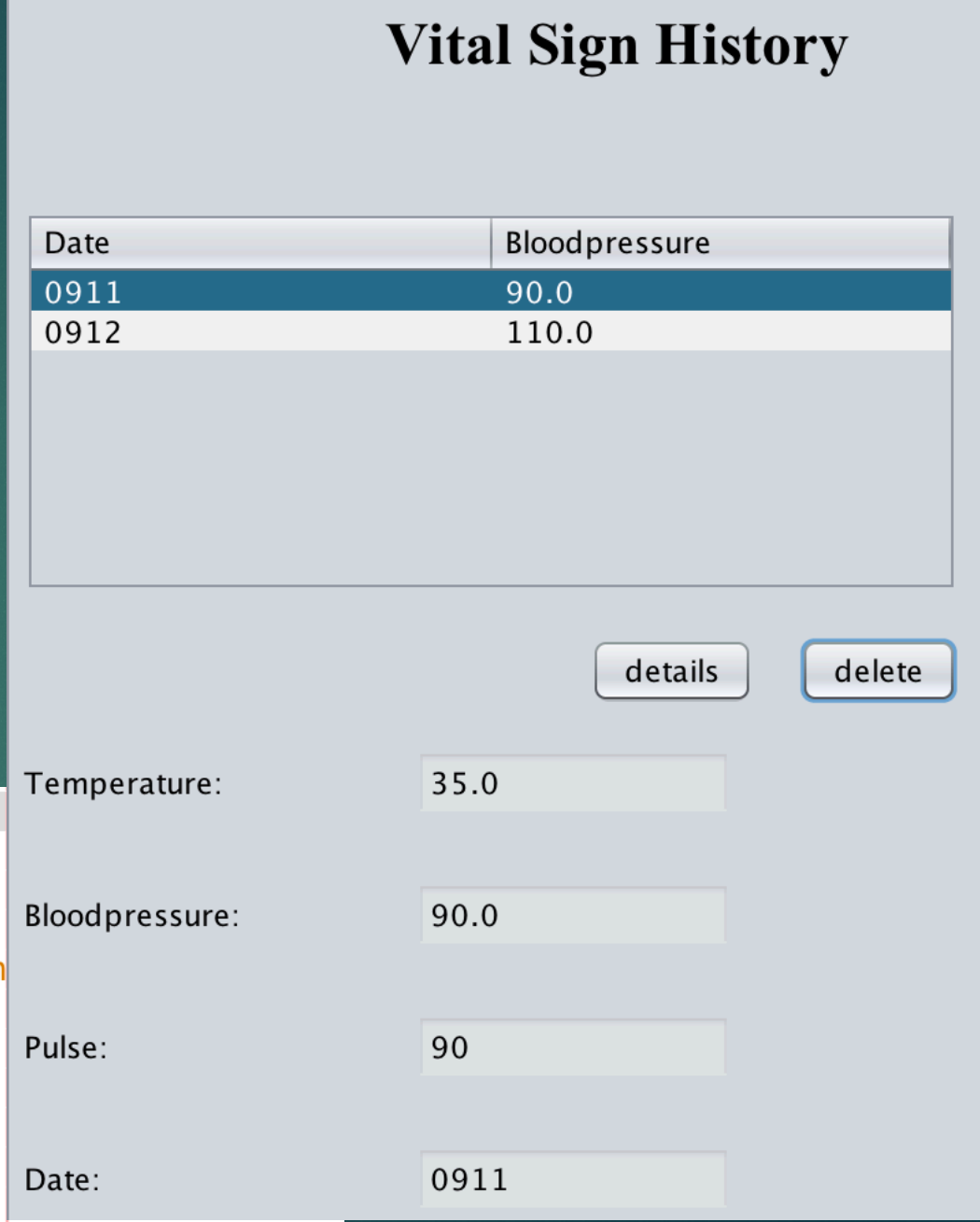
View Detail



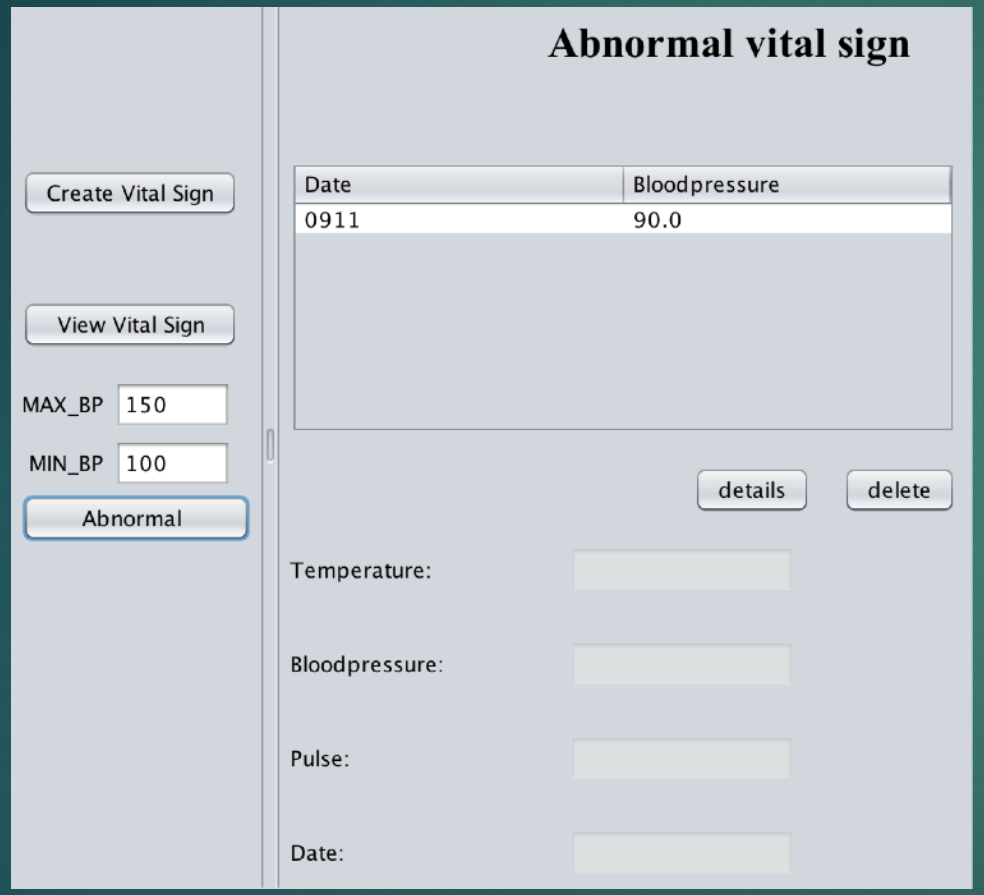

```
private void viewDetailActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int selectedRow = vitalSignTab.getSelectedRow();
    if (selectedRow>=0){
        VitalSign vs = (VitalSign) vitalSignTab.getValueAt(selectedRow, 0);
        bpTextField.setText(String.valueOf(vs.getBloodPressure()));
        dateTextField.setText(vs.getDate());
        pulseTextField.setText(String.valueOf(vs.getPulse()));
        tmpTextField.setText(String.valueOf(vs.getTemperature()));
    }else {
        JOptionPane.showMessageDialog(null, "Please select a specific vital sign", "Alert", JOptionPane.ERR
    }
```

Delete vital sign

```
private void deleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int selectedRow = vitalSignTab.getSelectedRow();
    if (selectedRow>=0){
        int dialogRes = JOptionPane.showConfirmDialog(null, "This vital sign
        if(dialogRes == JOptionPane.YES_OPTION){
        VitalSign vs = (VitalSign) vitalSignTab.getValueAt(selectedRow,0);
        vsh.delVitalSign(vs);
        populateTable();
        resetTextField();
        }
    }else{
        JOptionPane.showMessageDialog(null, "Please select a specific vital sign to delete", "
    }
```



Find the Abnormal Vital Signs



```java
private void abnBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String maxbpString = maxTxt.getText();
    String minbpString = minTxt.getText();
    double maxbp = maxbpString.equals("") ? Double.MAX_VALUE : Double.parseDouble(maxbpString);
    double minbp = minbpString.equals("") ? Double.MIN_VALUE : Double.parseDouble(minbpString);
    AbnormalPanel abnormalPanel = new AbnormalPanel(vsh, maxbp, minbp);
    jSplitPanel.setRightComponent(abnormalPanel);
}
```

Main JFrame

```java
private void populateTable() {
    DefaultTableModel dtm = (DefaultTableModel) vitalSignTab.getModel();
    dtm.setRowCount(0);
    for (VitalSign vs : vsh.getAbnormalList(maxbp, minbp)) {

        Object row[] = new Object[2];
        row[0] = vs;
        row[1] = vs.getBloodPressure();
        dtm.addRow(row);

    }

}
```

Abnormal Jpanel

```java
public List<VitalSign> getAbnormalList(double maxbp,double minbp){
    List<VitalSign> abnList = new ArrayList<>();

    for(VitalSign vs:vitalSignList){
        if(vs.getBloodPressure()>maxbp || vs.getBloodPressure()<minbp){
            abnList.add(vs);
        }
    }
    return abnList;
}
```

VitalsignHistory

Homework

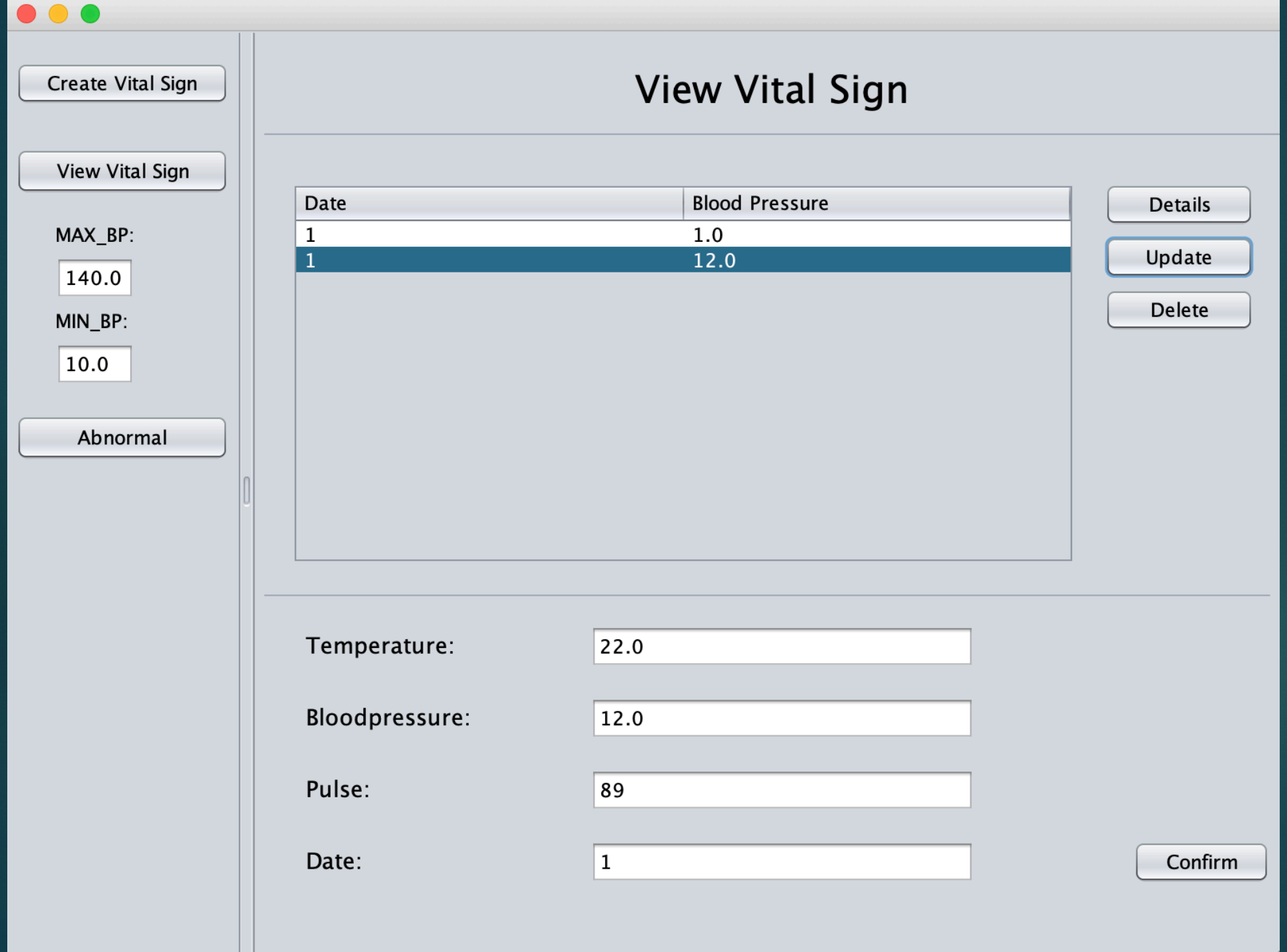
- Implement the functionality of finding the abnormal vital signs and display them. You could have your own criteria as default, but user should be able to modify the criteria.

  Default criteria example: blood pressure > 140 or blood pressure < 70 is abnormal.

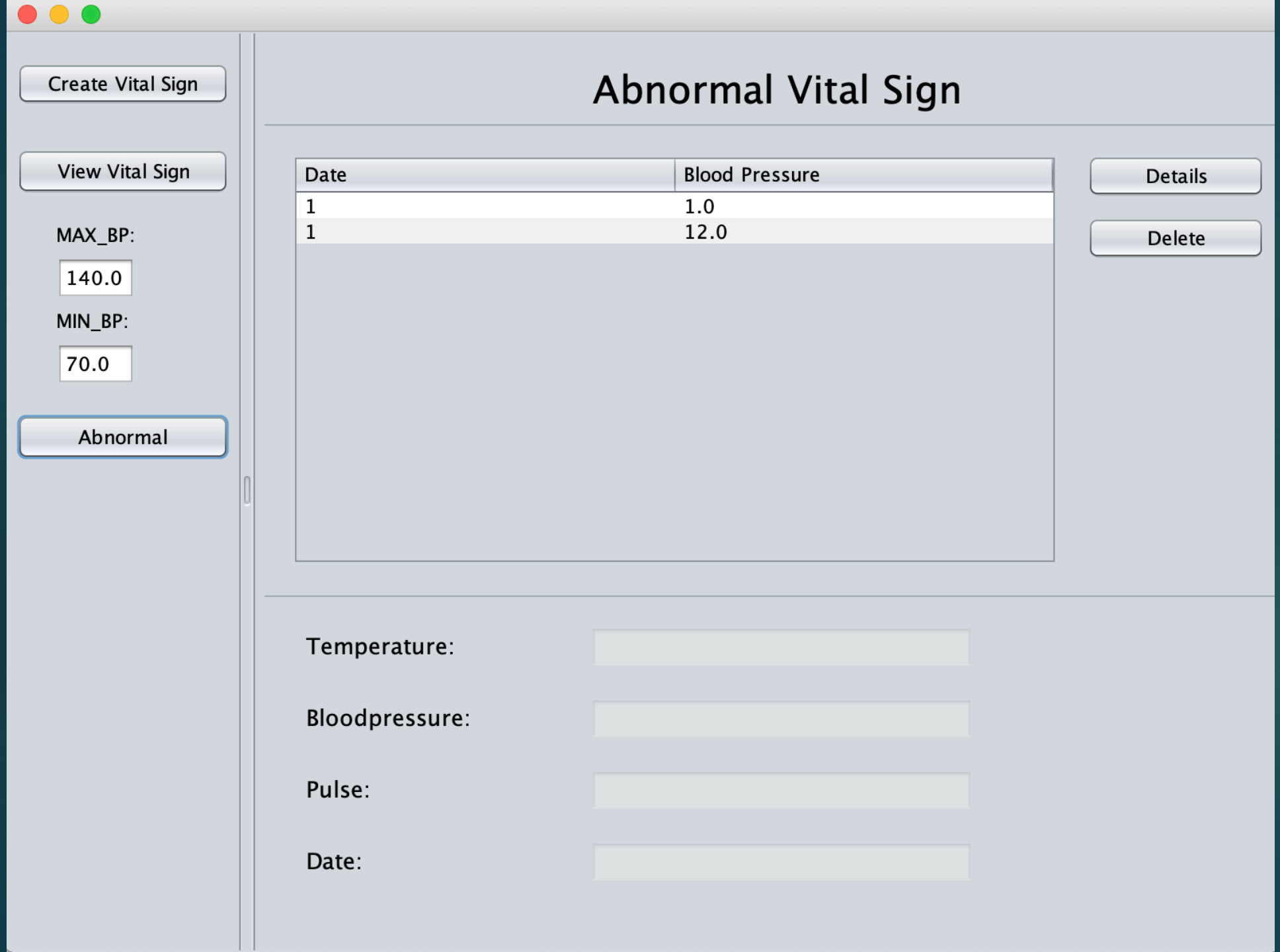
- Implement the functionality of updating vital signs on both "viewVitalJPanel" and "abnormalVitalJPanel".

- Multiple small git commits.

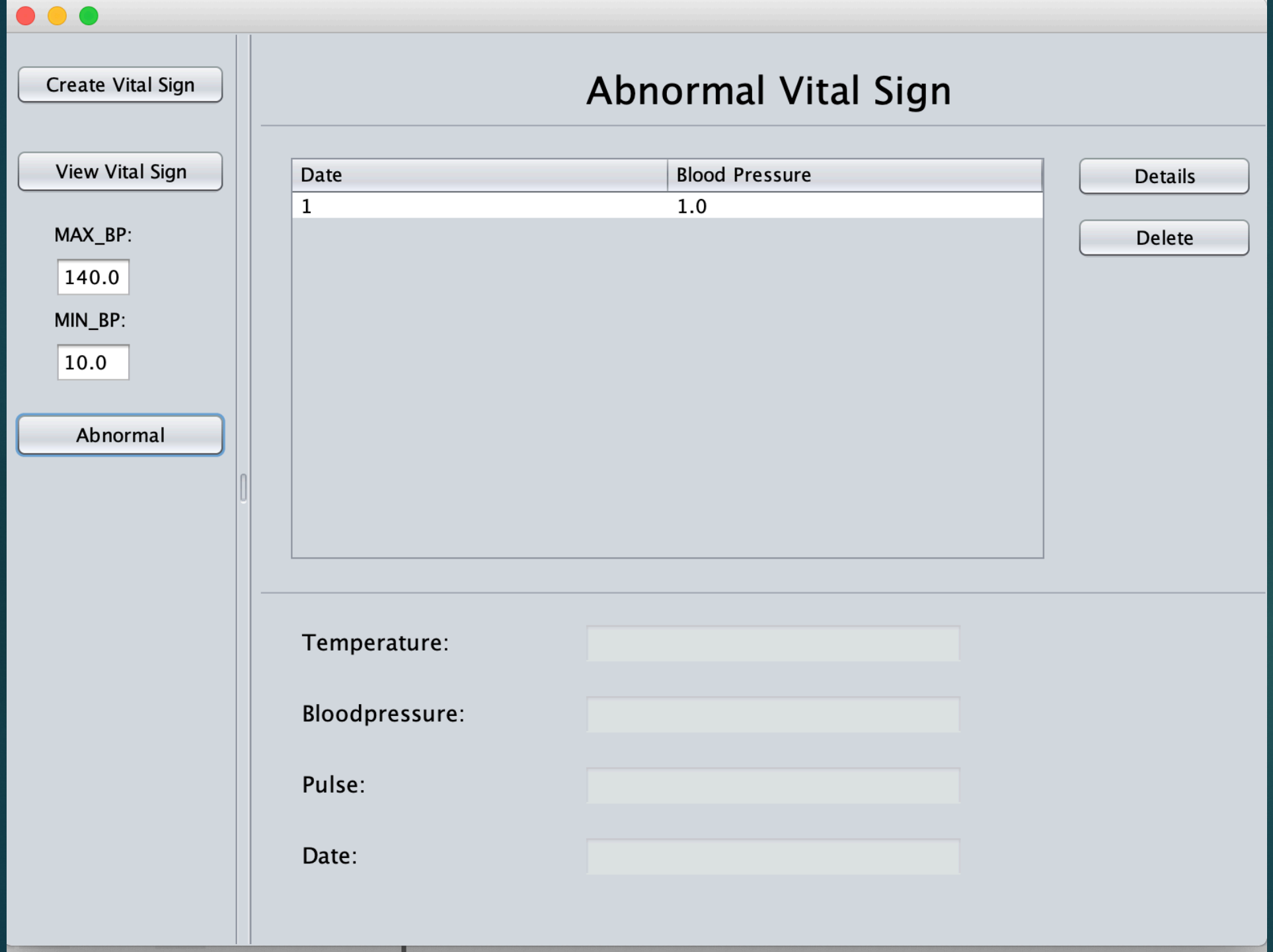
Due Date: Sat, Sep 28th, 11:59pm on BitBucket


Create Vital Sign
View Vital Sign
MAX_BP:
140.0
MIN_BP:
10.0
Abnormal
View Vital Sign
Date
Blood Pressure
1
1.0
1
12.0
Details
Update
Delete
Temperature:
22.0
Bloodpressure:
12.0
Pulse:
89
Date:
1
Confirm

Update


Create Vital Sign
View Vital Sign
MAX_BP:
140.0
MIN_BP:
70.0
Abnormal
Abnormal Vital Sign
Date
Blood Pressure
1
1.0
1
12.0
Details
Delete
Temperature:
Bloodpressure:
Pulse:
Date:

Abnormal


Create Vital Sign
View Vital Sign
MAX_BP:
140.0
MIN_BP:
10.0
Abnormal
Abnormal Vital Sign
Date
Blood Pressure
1
1.0
Details
Delete
Temperature:
Bloodpressure:
Pulse:
Date:

Abnormal