1. Project Setup:
Create a new Android project in Android Studio.

2. User Interface and Guided Flow:
Design the user interface using XML layout files. Create layouts for each test, instructions, and buttons for user interaction. Ensure that the user cannot proceed to the next test until the current one is completed.

Example layout for a camera test (camera_test_layout.xml):

```
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical">

   <TextView
       android:id="@+id/instructionTextView"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:text="Place your device in a well-lit area and ensure the
camera is not obstructed."
   />

   <Button
       android:id="@+id/startCameraTestButton"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:text="Start Camera Test"
   />
</LinearLayout>
```

3. Device Health Checks:
Create Java functions to perform device health checks for cameras, microphones, rooted status, Bluetooth, and sensors. Here's a simplified example for a camera check:

```
public class CameraTestActivity extends AppCompatActivity {
   private TextView instructionTextView;
   private Button startCameraTestButton;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.camera_test_layout);

       instructionTextView = findViewById(R.id.instructionTextView);
       startCameraTestButton = findViewById(R.id.startCameraTestButton);

       startCameraTestButton.setOnClickListener(new
View.OnClickListener() {
           @Override
           public void onClick(View view) {
               // Implement the camera test logic, provide instructions,
and update UI in real-time.
```

```
            }
        });
    }
}
```

4. Continuous Health Checks and Real-time UI Updates:

```
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class HealthCheckActivity extends Activity {
    private TextView instructionTextView;
    private Button nextButton;

    private State currentState;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_health_check);

        instructionTextView = findViewById(R.id.instructionTextView);
        nextButton = findViewById(R.id.nextButton);

        // Initialize the state machine with the first state
        currentState = new CameraTestState();

        nextButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Transition to the next state
                currentState = currentState.nextState();

                if (currentState != null) {
                    // Execute the current state's logic
                    currentState.performStateLogic();
                } else {
                    // All tests are completed
                    instructionTextView.setText("All tests completed.");
                }
            }
        });
    }
}
```

Now, define a State interface that all test states will implement:

```
public interface State {
    void performStateLogic();
    State nextState();
}
```

Create specific test states for each health check. Here's an example for the camera test:

```
public class CameraTestState implements State {
    @Override
    public void performStateLogic() {
        // Display camera test instructions to the user
        instructionTextView.setText("Place your device in a well-lit area
and ensure the camera is not obstructed.");
    }

    @Override
    public State nextState() {
        // Implement camera test logic, check for pass/fail

        // Transition to the next state (in this case, the microphone
test)
        return new MicrophoneTestState();
    }
}
```

5. Result Output Options:

Option 1: Send Data to Firebase

```
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

// Initialize Firebase
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference dataRef = database.getReference("healthCheckData");

// Create a data object to store the results
HealthCheckResult healthCheckResult = new HealthCheckResult();
healthCheckResult.setCameraStatus("Pass");
healthCheckResult.setMicrophoneStatus("Fail");
// Set other test results

// Push the data to Firebase
dataRef.push().setValue(healthCheckResult);
```

You would need to create a class HealthCheckResult to store the results of your health checks. You can then push this data to Firebase when the user chooses this option.

Option 2: Generate a PDF Report

```
import com.itextpdf.text.Document;
import com.itextpdf.text.DocumentException;
import com.itextpdf.text.Paragraph;
import com.itextpdf.text.pdf.PdfWriter;

// Create a new document
Document document = new Document();
try {
    PdfWriter.getInstance(document, new
FileOutputStream("health_check_report.pdf"));
    document.open();
```

```
    // Add content to the PDF
    document.add(new Paragraph("Health Check Report"));
    document.add(new Paragraph("Camera Status: Pass"));
    document.add(new Paragraph("Microphone Status: Fail"));
    // Add other test results

    document.close();
} catch (DocumentException | IOException e) {
    e.printStackTrace();
}
```