

```

import React, { useState, useEffect, useRef } from 'react';
import {
  Text,
  View,
  TextInput,
  TouchableOpacity,
  FlatList,
  ActivityIndicator,
  KeyboardAvoidingView,
  Platform,
  SafeAreaView
} from 'react-native';
import AsyncStorage from '@react-native-async-storage/async-storage';

const STORAGE_KEY = 'ncert_ai_tutor_messages';

export default function App() {
  const [question, setQuestion] = useState("");
  const [messages, setMessages] = useState([
    { id: 'm0', role: 'assistant', content: 'Hi! I am your NCERT AI Tutor. Ask me a question from Maths, Science, or SST.', timestamp: Date.now() }
  ]);
  const [loading, setLoading] = useState(false);
  const flatListRef = useRef(null);

  useEffect(() => {
    loadMessages();
  }, []);

  useEffect(() => {
    // Persist messages whenever they change
    AsyncStorage.setItem(STORAGE_KEY, JSON.stringify(messages)).catch(() => {});
    // Auto-scroll slightly after messages update
    if (flatListRef.current && messages.length > 0) {
      setTimeout(() => {
        flatListRef.current.scrollToEnd({ animated: true });
      }, 100);
    }
  }, [messages]);

  const loadMessages = async () => {
    try {
      const saved = await AsyncStorage.getItem(STORAGE_KEY);
      if (saved) setMessages(JSON.parse(saved));
    }
  }
}

```

```

    } catch (e) {
      // ignore load errors for now
    }
  };

const sendQuestion = async () => {
  const text = question.trim();
  if (!text || loading) return;

  const userMsg = { id: `u${Date.now()}`, role: 'user', content: text, timestamp: Date.now() };
  setMessages(prev => [...prev, userMsg]);
  setQuestion("");
  setLoading(true);

  try {
    // TODO: Replace generateMockReply with real API call (server proxy recommended)
    const aiReply = await fakeAiCall(text);
    const assistantMsg = { id: `a${Date.now()}`, role: 'assistant', content: aiReply, timestamp:
Date.now() };
    setMessages(prev => [...prev, assistantMsg]);
  } catch (err) {
    const errMsg = { id: `e${Date.now()}`, role: 'assistant', content: 'Sorry — there was a problem
getting the answer. Try again.', timestamp: Date.now() };
    setMessages(prev => [...prev, errMsg]);
  } finally {
    setLoading(false);
  }
};

// Placeholder for async AI call to allow easy swap with real API
const fakeAiCall

```