```cpp
#include <wx/wx.h>
#include <vector>
#include <string>

struct Account {
    int accountNumber;
    std::string name;
    std::string password;
    double balance;
    std::vector<std::string> history;
};

static std::vector<Account> accounts;
static int nextAccountNo = 2;   // admin is 1
static int loggedInIndex = -1;  // -1 = no one logged in

// ---------- helpers ----------
int findAccountIndexByNumber(int accNo) {
    for (size_t i = 0; i < accounts.size(); ++i) {
        if (accounts[i].accountNumber == accNo) return (int)i;
    }
    return -1;
}

wxString moneyFmt(double v) {
    return wxString::Format("KSh %.2f", v);
}

// ---------- History Window (separate) ----------
class HistoryFrame : public wxFrame {
public:
    HistoryFrame(wxWindow* parent, int accIndex)
        : wxFrame(parent, wxID_ANY, "Transaction History",
          wxDefaultPosition, wxSize(400, 300))
    {
        wxPanel* panel = new wxPanel(this);
        wxBoxSizer* s = new wxBoxSizer(wxVERTICAL);

        wxTextCtrl* box = new wxTextCtrl(panel, wxID_ANY, "",
          wxDefaultPosition, wxDefaultSize,
            wxTE_MULTILINE | wxTE_READONLY);
        if (accIndex >= 0 && accIndex < (int)accounts.size()) {
            for (const auto& h : accounts[accIndex].history) {
                box->AppendText(h + "\n");
            }
        }
        else {
            box->SetValue("No history available.");
        }
```

```
48
49          s->Add(box, 1, wxEXPAND | wxALL, 8);
50          panel->SetSizer(s);
51      }
52  };
53
54  // ---------- Main Window (login + dashboard with dynamic center) ----------
55  class MainFrame : public wxFrame {
56  public:
57      MainFrame()
58          : wxFrame(NULL, wxID_ANY, "Simple Banking App",
              wxDefaultPosition, wxSize(420, 420))
59      {
60          wxPanel* panel = new wxPanel(this);
61          mainPanel = panel;
62
63          mainSizer = new wxBoxSizer(wxVERTICAL);
64
65          // Top: Balance label (hidden until login)
66          balanceLabel = new wxStaticText(panel, wxID_ANY, "");
67          balanceLabel->SetFont(wxFont(14, wxFONTFAMILY_DEFAULT,
              wxFONTSTYLE_NORMAL, wxFONTWEIGHT_BOLD));
68          balanceLabel->Hide();
69          mainSizer->Add(balanceLabel, 0, wxALIGN_CENTER | wxTOP |
              wxBOTTOM, 8);
70
71          // Middle: dynamic content area
72          contentPanel = new wxPanel(panel, wxID_ANY, wxDefaultPosition,
              wxSize(380, 240));
73          contentSizer = new wxBoxSizer(wxVERTICAL);
74          contentPanel->SetSizer(contentSizer);
75          mainSizer->Add(contentPanel, 1, wxEXPAND | wxLEFT | wxRIGHT, 8);
76
77          // Bottom: button bar (dashboard actions) - hidden until login
78          buttonBar = new wxPanel(panel, wxID_ANY);
79          wxBoxSizer* bbs = new wxBoxSizer(wxHORIZONTAL);
80          btnDeposit = new wxButton(buttonBar, wxID_ANY, "Deposit");
81          btnWithdraw = new wxButton(buttonBar, wxID_ANY, "Withdraw");
82          btnTransfer = new wxButton(buttonBar, wxID_ANY, "Transfer");
83          btnHistory = new wxButton(buttonBar, wxID_ANY, "History");
84          btnLogout = new wxButton(buttonBar, wxID_ANY, "Logout");
85
86          bbs->Add(btnDeposit, 1, wxALL | wxEXPAND, 4);
87          bbs->Add(btnWithdraw, 1, wxALL | wxEXPAND, 4);
88          bbs->Add(btnTransfer, 1, wxALL | wxEXPAND, 4);
89          bbs->Add(btnHistory, 1, wxALL | wxEXPAND, 4);
90          bbs->Add(btnLogout, 1, wxALL | wxEXPAND, 4);
91          buttonBar->SetSizer(bbs);
```

```
92          buttonBar->Hide();
93          mainSizer->Add(buttonBar, 0, wxEXPAND | wxLEFT | wxRIGHT |
              wxBOTTOM, 8);
94
95          panel->SetSizer(mainSizer);
96
97          // Bind buttons
98          btnDeposit->Bind(wxEVT_BUTTON, &MainFrame::OnDepositClicked,
              this);
99          btnWithdraw->Bind(wxEVT_BUTTON, &MainFrame::OnWithdrawClicked,
              this);
100         btnTransfer->Bind(wxEVT_BUTTON, &MainFrame::OnTransferClicked,
              this);
101         btnHistory->Bind(wxEVT_BUTTON, &MainFrame::OnHistoryClicked,
              this);
102         btnLogout->Bind(wxEVT_BUTTON, &MainFrame::OnLogoutClicked, this);
103
104         // Start app with initial view (create / login / exit on same
              window)
105         ShowInitialView();
106     }
107
108 private:
109     wxPanel* mainPanel = nullptr;
110     wxBoxSizer* mainSizer = nullptr;
111
112     wxStaticText* balanceLabel = nullptr;
113     wxPanel* contentPanel = nullptr;
114     wxBoxSizer* contentSizer = nullptr;
115
116     wxPanel* buttonBar = nullptr;
117     wxButton* btnDeposit = nullptr;
118     wxButton* btnWithdraw = nullptr;
119     wxButton* btnTransfer = nullptr;
120     wxButton* btnHistory = nullptr;
121     wxButton* btnLogout = nullptr;
122
123     // Clears center content area
124     void ClearContent() {
125         contentPanel->Freeze();
126         contentPanel->DestroyChildren();
127         contentSizer = new wxBoxSizer(wxVERTICAL);
128         contentPanel->SetSizer(contentSizer);
129         contentPanel->Layout();
130         contentPanel->Thaw();
131     }
132
133     // ---- Initial (login/create/exit) UI ----
134     void ShowInitialView() {
```

```
135         loggedInIndex = -1;
136         balanceLabel->Hide();
137         buttonBar->Hide();
138         ClearContent();
139 
140         // Big title
141         wxStaticText* title = new wxStaticText(contentPanel, wxID_ANY, "
              Welcome — Simple Bank", wxDefaultPosition, wxDefaultSize);
142         title->SetFont(wxFont(12, wxFONTFAMILY_DEFAULT,
              wxFONTSTYLE_NORMAL, wxFONTWEIGHT_BOLD));
143         contentSizer->Add(title, 0, wxALIGN_CENTER | wxTOP, 12);
144 
145         // Create Account button
146         wxButton* createBtn = new wxButton(contentPanel, wxID_ANY, "
              Create Account");
147         contentSizer->Add(createBtn, 0, wxEXPAND | wxALL, 10);
148         createBtn->Bind(wxEVT_BUTTON, &MainFrame::OnCreateAccount, this);
149 
150         // Login button
151         wxButton* loginBtn = new wxButton(contentPanel, wxID_ANY,
              "Login");
152         contentSizer->Add(loginBtn, 0, wxEXPAND | wxALL, 10);
153         loginBtn->Bind(wxEVT_BUTTON, &MainFrame::OnLoginStart, this);
154 
155         // Exit button
156         wxButton* exitBtn = new wxButton(contentPanel, wxID_ANY, "Exit");
157         contentSizer->Add(exitBtn, 0, wxEXPAND | wxALL, 10);
158         exitBtn->Bind(wxEVT_BUTTON, [](wxCommandEvent&) { wxExit(); });
159 
160         contentPanel->Layout();
161         mainPanel->Layout();
162     }
163 
164     // ---- Dashboard view (balance + buttonBar) ----
165     void ShowDashboardView() {
166         ClearContent();
167         if (loggedInIndex < 0 || loggedInIndex >= (int)accounts.size()) {
168             wxMessageBox("Invalid logged in account.");
169             ShowInitialView();
170             return;
171         }
172 
173         // Show balance label
174         balanceLabel->SetLabel("Balance: " + moneyFmt(accounts
              [loggedInIndex].balance));
175         balanceLabel->Show();
176 
177         // Welcome text
178         wxString w = "Welcome, " + wxString(accounts[loggedInIndex].name);
```

```
179         wxStaticText* welcome = new wxStaticText(contentPanel, wxID_ANY,
              w);
180         welcome->SetFont(wxFont(11, wxFONTFAMILY_DEFAULT,
              wxFONTSTYLE_NORMAL, wxFONTWEIGHT_NORMAL));
181         contentSizer->Add(welcome, 0, wxALIGN_LEFT | wxTOP | wxLEFT, 8);
182
183         // Placeholder center content
184         wxStaticText* hint = new wxStaticText(contentPanel, wxID_ANY,
              "Choose an action below. The main area will change like a
              mobile app screen.");
185         contentSizer->Add(hint, 0, wxEXPAND | wxALL, 8);
186
187         buttonBar->Show();
188         contentPanel->Layout();
189         mainPanel->Layout();
190     }
191
192     // ---- Handlers for Initial UI ----
193     void OnCreateAccount(wxCommandEvent& evt) {
194         wxTextEntryDialog nameDlg(this, "Enter your name:", "Create
              Account");
195         if (nameDlg.ShowModal() != wxID_OK) return;
196         wxString name = nameDlg.GetValue();
197
198         wxTextEntryDialog passDlg(this, "Set a password:", "Create
              Account");
199         if (passDlg.ShowModal() != wxID_OK) return;
200         wxString pass = passDlg.GetValue();
201
202         Account acc;
203         acc.accountNumber = nextAccountNo++;
204         acc.name = std::string(name.mb_str());
205         acc.password = std::string(pass.mb_str());
206         acc.balance = 0.0;
207         acc.history.push_back("Account created with balance 0");
208         accounts.push_back(acc);
209
210         wxMessageBox("☑ Account created!\nYour Account Number is " +
              wxString::Format("%d", acc.accountNumber));
211     }
212
213     void OnLoginStart(wxCommandEvent& evt) {
214         wxTextEntryDialog accDlg(this, "Enter account number:", "Login");
215         if (accDlg.ShowModal() != wxID_OK) return;
216         long long accNoLL = 0;
217         accDlg.GetValue().ToLongLong(&accNoLL);
218         int accNo = (int)accNoLL;
219
220         wxTextEntryDialog passDlg(this, "Enter password:", "Login");
```

```
221         if (passDlg.ShowModal() != wxID_OK) return;
222         wxString pass = passDlg.GetValue();
223 
224         int idx = findAccountIndexByNumber(accNo);
225         if (idx >= 0 && accounts[idx].password == std::string(pass.mb_str
              ())) {
226             loggedInIndex = idx;
227             wxMessageBox("☑ Login successful! Welcome " + accounts
                [loggedInIndex].name);
228             ShowDashboardView();
229         }
230         else {
231             wxMessageBox("❌ Login failed. Wrong account number or
                password.");
232         }
233     }
234 
235     // ---- Dashboard button handlers (replace central panel content) ----
236     void OnDepositClicked(wxCommandEvent& evt) {
237         ClearContent();
238         wxStaticText* w = new wxStaticText(contentPanel, wxID_ANY,
              "Deposit Money");
239         w->SetFont(wxFont(11, wxFONTFAMILY_DEFAULT, wxFONTSTYLE_NORMAL,
              wxFONTWEIGHT_BOLD));
240         contentSizer->Add(w, 0, wxLEFT | wxTOP, 8);
241 
242         wxStaticText* amtLabel = new wxStaticText(contentPanel, wxID_ANY,
              "Amount:");
243         contentSizer->Add(amtLabel, 0, wxLEFT | wxTOP, 10);
244 
245         wxTextCtrl* amtCtrl = new wxTextCtrl(contentPanel, wxID_ANY);
246         contentSizer->Add(amtCtrl, 0, wxLEFT | wxRIGHT | wxEXPAND, 10);
247 
248         wxButton* confirm = new wxButton(contentPanel, wxID_ANY,
              "Confirm");
249         contentSizer->Add(confirm, 0, wxALL | wxALIGN_CENTER, 10);
250 
251         confirm->Bind(wxEVT_BUTTON, [=](wxCommandEvent&) {
252             if (loggedInIndex < 0) { wxMessageBox("Not logged in.");
                return; }
253             double v;
254             if (!amtCtrl->GetValue().ToDouble(&v) || v <= 0) {
255                 wxMessageBox("Invalid amount.");
256                 return;
257             }
258             accounts[loggedInIndex].balance += v;
259             accounts[loggedInIndex].history.push_back("Deposited " +
                std::to_string(v));
260             wxMessageBox("☑ Deposit successful.\nNew balance: " +
```

```
                moneyFmt(accounts[loggedInIndex].balance));
261             // refresh balance label
262             balanceLabel->SetLabel("Balance: " + moneyFmt(accounts
                  [loggedInIndex].balance));
263             // go back to dashboard main content
264             ShowDashboardView();
265             });
266
267         contentPanel->Layout();
268         mainPanel->Layout();
269     }
270
271     void OnWithdrawClicked(wxCommandEvent& evt) {
272         ClearContent();
273         wxStaticText* w = new wxStaticText(contentPanel, wxID_ANY,
              "Withdraw Money");
274         w->SetFont(wxFont(11, wxFONTFAMILY_DEFAULT, wxFONTSTYLE_NORMAL,
              wxFONTWEIGHT_BOLD));
275         contentSizer->Add(w, 0, wxLEFT | wxTOP, 8);
276
277         wxStaticText* amtLabel = new wxStaticText(contentPanel, wxID_ANY,
              "Amount:");
278         contentSizer->Add(amtLabel, 0, wxLEFT | wxTOP, 10);
279
280         wxTextCtrl* amtCtrl = new wxTextCtrl(contentPanel, wxID_ANY);
281         contentSizer->Add(amtCtrl, 0, wxLEFT | wxRIGHT | wxEXPAND, 10);
282
283         wxButton* confirm = new wxButton(contentPanel, wxID_ANY,
              "Confirm");
284         contentSizer->Add(confirm, 0, wxALL | wxALIGN_CENTER, 10);
285
286         confirm->Bind(wxEVT_BUTTON, [=](wxCommandEvent&) {
287             if (loggedInIndex < 0) { wxMessageBox("Not logged in.");
                  return; }
288             double v;
289             if (!amtCtrl->GetValue().ToDouble(&v) || v <= 0) {
290                 wxMessageBox("Invalid amount.");
291                 return;
292             }
293             if (v > accounts[loggedInIndex].balance) {
294                 wxMessageBox("❌ Insufficient funds.");
295                 accounts[loggedInIndex].history.push_back("Failed
                      withdrawal " + std::to_string(v));
296                 return;
297             }
298             accounts[loggedInIndex].balance -= v;
299             accounts[loggedInIndex].history.push_back("Withdrew " +
                  std::to_string(v));
300             wxMessageBox("☑ Withdraw successful.\nRemaining: " + moneyFmt
```

```
                (accounts[loggedInIndex].balance));
301             balanceLabel->SetLabel("Balance: " + moneyFmt(accounts
                  [loggedInIndex].balance));
302             ShowDashboardView();
303             });
304
305         contentPanel->Layout();
306         mainPanel->Layout();
307     }
308
309     void OnTransferClicked(wxCommandEvent& evt) {
310         ClearContent();
311         wxStaticText* w = new wxStaticText(contentPanel, wxID_ANY,
              "Transfer Money");
312         w->SetFont(wxFont(11, wxFONTFAMILY_DEFAULT, wxFONTSTYLE_NORMAL,
              wxFONTWEIGHT_BOLD));
313         contentSizer->Add(w, 0, wxLEFT | wxTOP, 8);
314
315         contentSizer->Add(new wxStaticText(contentPanel, wxID_ANY, "
              Target Account No:"), 0, wxLEFT | wxTOP, 8);
316         wxTextCtrl* targetCtrl = new wxTextCtrl(contentPanel, wxID_ANY);
317         contentSizer->Add(targetCtrl, 0, wxLEFT | wxRIGHT | wxEXPAND, 10);
318
319         contentSizer->Add(new wxStaticText(contentPanel, wxID_ANY,
              "Amount:"), 0, wxLEFT | wxTOP, 8);
320         wxTextCtrl* amtCtrl = new wxTextCtrl(contentPanel, wxID_ANY);
321         contentSizer->Add(amtCtrl, 0, wxLEFT | wxRIGHT | wxEXPAND, 10);
322
323         wxButton* confirm = new wxButton(contentPanel, wxID_ANY,
              "Confirm");
324         contentSizer->Add(confirm, 0, wxALL | wxALIGN_CENTER, 10);
325
326         confirm->Bind(wxEVT_BUTTON, [=](wxCommandEvent&) {
327             if (loggedInIndex < 0) { wxMessageBox("Not logged in.");
                  return; }
328             long long targetLL = 0;
329             if (!targetCtrl->GetValue().ToLongLong(&targetLL))
                  { wxMessageBox("Invalid account number."); return; }
330             int target = (int)targetLL;
331             double v;
332             if (!amtCtrl->GetValue().ToDouble(&v) || v <= 0)
                  { wxMessageBox("Invalid amount."); return; }
333
334             if (v > accounts[loggedInIndex].balance) {
335                 wxMessageBox("❌ Insufficient funds.");
336                 accounts[loggedInIndex].history.push_back("Failed
                      transfer " + std::to_string(v) + " to " + std::to_string
                      (target));
337                 return;
```

```
338             }
339             int idx = findAccountIndexByNumber(target);
340             if (idx < 0) { wxMessageBox("❌ Target account not found.");
                  return; }
341             if (idx == loggedInIndex) { wxMessageBox("Cannot transfer to
                  same account."); return; }
342
343             accounts[loggedInIndex].balance -= v;
344             accounts[idx].balance += v;
345             accounts[loggedInIndex].history.push_back("Transferred " +
                  std::to_string(v) + " to " + accounts[idx].name);
346             accounts[idx].history.push_back("Received " + std::to_string
                  (v) + " from " + accounts[loggedInIndex].name);
347             wxMessageBox("☑ Transferred " + wxString::Format("%.2f",
                  v) + " to " + accounts[idx].name);
348             balanceLabel->SetLabel("Balance: " + moneyFmt(accounts
                  [loggedInIndex].balance));
349             ShowDashboardView();
350             });
351
352         contentPanel->Layout();
353         mainPanel->Layout();
354     }
355
356     void OnHistoryClicked(wxCommandEvent& evt) {
357         if (loggedInIndex < 0) { wxMessageBox("Not logged in."); return; }
358         HistoryFrame* h = new HistoryFrame(this, loggedInIndex);
359         h->Show();
360     }
361
362     void OnLogoutClicked(wxCommandEvent& evt) {
363         loggedInIndex = -1;
364         ShowInitialView();
365     }
366 };
367
368 // ---------- App ----------
369 class MyApp : public wxApp {
370 public:
371     virtual bool OnInit() {
372         // default Admin
373         Account admin;
374         admin.accountNumber = 1;
375         admin.name = "Admin";
376         admin.password = "1234";
377         admin.balance = 2000.0;
378         admin.history.push_back("Admin account created with balance
              2000");
379         accounts.push_back(admin);
```

```
380
381         MainFrame* f = new MainFrame();
382         f->Show();
383         return true;
384     }
385 };
386
387 wxIMPLEMENT_APP(MyApp);
388
```