



Thoughtful project design is one of the best ways to improve data quality, reduce rework, and make your study easier to manage.

Use these best practices when planning, building, testing, securing, and maintaining your REDCap project:

Plan First. Plan Before You Build. Take time to define your project requirements before creating instruments in REDCap. Consider the following:

- What data do you need to collect?
- Why is each data element needed?
- Who will enter the data?
- Will participants complete surveys?
- Will data be collected once, repeatedly, or longitudinally?
- Will multiple sites or groups be involved?
- What reports, exports, or analyses will be needed?

Best Practice: Create a data collection plan or data dictionary before building your project.

Design Clearly. Use Clear and Consistent Variable Names. Good variable names make projects easier to understand, maintain, and analyze.

- Use lowercase letters.
- Use underscores instead of spaces.
- Keep names short but descriptive.
- Establish and follow a consistent naming convention.
- Avoid changing variable names after data collection begins.

Best Practice: Choose variable names carefully before collecting data.

Test Thoroughly. Use Field Validation Whenever Possible. Use REDCap's validation features to improve data quality and prevent common entry errors:

- Use date validation for dates.
- Use numeric validation for numbers.
- Use email validation for email addresses.
- Set appropriate ranges or limits when applicable.

Best Practice: Use structured fields and validation instead of free-text fields whenever possible.

Use Branching Logic Carefully. Branching logic can simplify data collection by displaying fields only when they are relevant.

- Keep the logic as simple as possible.
- Make sure conditions are based on the correct variables and values.
- Test every possible response path.
- Review the logic when modifying related fields.

Best Practice: Test branching logic using realistic scenarios before moving the project to Production.

Use Calculated Fields Sparingly. Calculated fields can automate simple calculations, but excessive use can make a project more difficult to maintain. Before adding a calculation:

- Confirm that the calculation is necessary.
- Use the simplest formula possible.
- Make sure the underlying fields are appropriate for the calculation.
- Test the calculation with different values and scenarios.

Best Practice: Use calculated fields when they provide clear value, but avoid unnecessary complexity.

Write Clear and Descriptive Field Labels. Field labels should be easy for users and participants to understand:

- Use plain, concise language.
- Avoid unnecessary jargon and abbreviations.
- Provide instructions when a question may be unclear.
- Keep wording consistent throughout the project.
- Make response options clear and mutually understandable.

Best Practice: Have someone unfamiliar with the project review your instruments. If they cannot easily understand what to enter or select, revise the wording.

Design Instruments for Data Entry and Data Quality. Design forms and surveys with the end user in mind:

- Use appropriate field types.
- Group related questions together.
- Use clear instructions when needed.

- Minimize unnecessary data entry.
- Use required fields when a response is essential.
- Use validation and branching logic to prevent errors and reduce unnecessary questions.

Best Practice: Design instruments so users can enter accurate data efficiently and consistently.

Assign User Rights Carefully. Give users only the access they need to perform their responsibilities:

- Follow the principle of least privilege.
- Use user roles and Data Access Groups when appropriate.
- Limit access to sensitive data.
- Limit data export privileges.
- Remove access when users no longer need it.
- Review user access periodically.

Important: Users with access to User Rights can modify permissions for other users. Assign this privilege carefully.

Use Development and Production Properly. Build and test projects in Development before moving them to Production. Avoid making untested changes directly in Production, particularly after data collection has begun. Before moving a project to Production:

- Complete testing.
- Verify instruments and field settings.
- Confirm branching logic and calculations.
- Test surveys and automated notifications.
- Review user permissions.
- Verify reports and exports.

Best Practice: Treat Production as the live study environment and make changes carefully.

Test Your Project Thoroughly. Testing should simulate how the project will actually be used and should include the following:

- Data entry
- Branching logic
- Calculated fields
- Surveys and invitations
- Automated notifications
- Reports
- Data exports

- User roles and permissions
- Different response scenarios

Best Practice: Use realistic test cases and test every major workflow before moving the project to Production.

Document Your Project. Good documentation makes projects easier to maintain and transition between team members. Document important decisions, including:

- The purpose of the project.
- Data collection procedures.
- Naming conventions.
- Important branching logic and calculations.
- User roles and access requirements.
- Data quality procedures.
- Reporting and export requirements.
- Significant project changes.

Best Practice: Keep project documentation current throughout the life of the project—not just during initial development.

Manage Your Project After Go-Live. REDCap projects require ongoing management after they move to Production. Regularly:

- Run Data Quality checks.
- Review project activity and logging.
- Monitor reports and data exports.
- Review user access and permissions.
- Use record locking when appropriate.
- Address data quality issues promptly.
- Archive the project when it is complete.

Best Practice: Project owners are responsible for maintaining appropriate data quality, security, access, and documentation throughout the project lifecycle.