#### Database Setup:

Set up your database, considering the schema required to store information related to MVPs, collections, forms, users, and permissions. If you're using MongoDB, create collections and define their structures.

#### User Authentication and Authorization:

Implement user authentication and authorization using a library or framework like Passport.js. Define user roles and permissions for managing MVPs, collections, and forms.

#### API Design and Development:

Design your RESTful API endpoints for creating, reading, updating, and deleting MVPs, collections, forms, and user interactions. Develop API routes and controllers.

#### User Interface (UI) Development:

Begin building the user interface by creating the initial set of components and pages required for user registration, MVP creation, and user dashboard management.

#### User Registration and Login:

Implement user registration and login functionality. Use JSON Web Tokens (JWT) or an authentication method of your choice to secure user sessions.

#### MVP Creation:

Develop the MVP creation module, allowing users to create and customize MVPs with unique names. Store MVP-related data in the database.

#### MVP Forms and Collections:

Implement the creation of MVP forms and define collections. Develop features for CRUD operations on collections and MVP forms. Ensure that permissions are correctly applied.

#### User Collaboration:

For user collaboration on MVPs, set up real-time features using technologies like WebSockets (e.g., Socket.io). Allow multiple users to work on the same MVP while enforcing permissions.

#### Custom Views and Components:

Build the functionality to create and customize views using predefined components such as text, headings, tables, forms, and buttons. Enable interactive actions like creating, updating, or deleting collections data.

#### User Roles and Permissions:

Implement user roles and fine-grained permissions, allowing MVP creators to define access levels for other users interacting with their MVPs.

#### Testing:

Write unit tests and integration tests for your application to ensure that it functions correctly and handles user interactions securely.

#### Deployment:

Prepare your application for deployment by configuring production settings and optimizing performance. Deploy the application to a hosting platform like AWS, Azure, or Heroku.

#### Continuous Integration and Deployment (CI/CD):

Set up CI/CD pipelines to automate the build, testing, and deployment processes.

#### Monitoring and Logging:

Implement logging and monitoring solutions to keep track of application usage and troubleshoot any issues.

#### Documentation:

Create documentation for your project, including API documentation, user guides, and code documentation to assist other developers and users.

#### User Acceptance Testing (UAT):

Conduct user acceptance testing with a group of users or stakeholders to identify and address any issues before launching the MVP.

#### Launch and Deployment:

Deploy your MVP to a production environment and make it accessible to your target audience.

#### Feedback and Iteration:

Gather user feedback and iterate on the application to improve and expand its features based on user needs.

#### Marketing and Promotion:

Plan marketing and promotion strategies to attract users to your MVP builder platform.