**Project Description and Requirements**

**1.1 Project Description**

Develop a Todo website using Firebase, Next.js, and Git technologies. The project should include functionalities for searching, creating, reading, updating, and deleting (CRUD) todos through a REST-API service, using Firebase Firestore Database.

**1.2 Project Requirements**

**Front-end Development**

1. **User Authentication:**
   * Implement login and registration screens.
   * Ensure persistent login sessions (user should not have to log in again after the first time).
   * Gather e-mail, password, and name-surname during registration.
   * Password requirements: minimum 8 characters, with email validation.
2. **User Session Management:**
   * Enable users to terminate their sessions (log out functionality).
3. **Todo Management:**
   * Display two tabs in the navbar: “Completed” and “Incomplete”.
   * Sort todos by creation date (new to old) on the server-side.
   * Implement search functionality at the top of both “Completed” and “Incomplete” todos pages, searching by todo content.
   * Ensure that completing an incomplete todo moves it out of the Incomplete list.
   * Design flexible forms for adding, updating, and deleting todos with appropriate form validations using react-hook-form.
4. **Design and Responsiveness:**
   * Implement responsive design using Bootstrap.
   * Follow the Atomic Design Pattern for organizing UI components.
5. **Caching:**
   * Cache todos using Redis and update the cache on any CRUD operations.
6. **Browser Tab Customization:**
   * Set the browser tab image and title appropriately.

**Backend Development and Firebase Integration**

1. **Database Management:**
   * Use Firebase Firestore for managing all data operations.
   * Create a Firebase project and integrate it with the web app using Next.js and preferably TypeScript.
   * Store sensitive data (e.g., API Key, Project ID) in a .env file.
   * Implement a REST-API with Next.js for CRUD operations.
   * Maintain a user authentication table in Firestore, storing name and surname in a single variable and the account creation date.
   * Use email/password as the sign-in method.

**Version Control and Development Process**

1. **Git Usage:**
   * Use Git for version control (preferably Bitbucket or GitHub).
   * Commit changes incrementally with meaningful commit messages.
   * Establish a branch structure with feature branches and merge via pull requests to the main branch (bonus).
   * Follow clean coding practices and add comment lines where necessary (bonus).
   * Handle loading states, errors, and empty data returns from the database (bonus).
   * Research and understand unit testing tools and practices (bonus).
   * Deploy the project using platforms like Vercel or GitHub (bonus).

**Step-by-Step Implementation**

**Step 1: Set Up the Project**

* Initialize a Next.js project.
* Set up Firebase and Firestore.
* Configure environment variables for sensitive data.

**Step 2: Implement Authentication**

* Create registration and login pages.
* Implement form validations and email verification.
* Set up Firebase authentication with email/password.

**Step 3: Develop Todo Management Features**

* Create pages for “Completed” and “Incomplete” todos.
* Implement server-side sorting of todos.
* Add search functionality.
* Create forms for adding, updating, and deleting todos.
* Implement form validations using react-hook-form.

**Step 4: Implement Caching with Redis**

* Set up Redis for caching todos.
* Update cache on CRUD operations.

**Step 5: Enhance User Experience**

* Ensure responsive design using Bootstrap.
* Apply Atomic Design Pattern.
* Customize browser tab image and title.

**Step 6: Version Control**

* Regularly commit changes with meaningful messages.
* Use feature branches and pull requests for merging.
* Follow clean coding practices and add necessary comments.

**Step 7: Deployment -- Optional**

* Deploy the project on platforms like Vercel.
* Ensure the deployment is smooth and the application works as expected.