Project Report(Advanced URL Shortener)

The development of the Advanced URL Shortener Web Application was carried out in a systematic and structured manner to ensure that all requirements were met efficiently.

Understanding the Problem:
The first step was to clearly understand the project requirements. The application needed to shorten long URLs and allow users to store and manage them securely. Unlike a basic URL shortener, this project required user authentication so that each user could maintain a personal URL history.

Planning the Application Flow:
After understanding the requirements, the overall workflow was planned. The flow begins with user registration and login. Once logged in, the user is redirected to a dashboard where URLs can be shortened. Each shortened URL is saved in the database and displayed only to the respective user.

Database Design:
Two database tables were designed. The User table stores user credentials, while the URL table stores the original URL, shortened URL, and user reference. A foreign key relationship links URLs to users, ensuring user-specific data storage.

User Authentication:
User authentication was implemented using Flask-Login. Signup validates username length and uniqueness, while login verifies credentials and manages user sessions. Access to the dashboard is restricted to logged-in users only.

URL Shortening Logic:
A random alphanumeric code is generated for each URL. Before saving, the URL format is validated. The original URL and its short code are then stored in the database.

URL Redirection:
When a shortened URL is accessed, the application searches for the short code in the database and redirects the user to the original URL.

User Dashboard:

The dashboard allows users to shorten new URLs and view their URL history. Each user can see only their own URLs, ensuring privacy.

Validation and Testing:

Input validation and error handling were implemented to prevent incorrect data. The application was tested by creating multiple users, shortening URLs, and verifying correct redirection.

Conclusion:

The solution was approached step by step, starting from requirement analysis to testing. Breaking the problem into smaller modules made development efficient and the final application meets all project requirements.