Cloud-Based To-Do List Application with Serverless Backend

A Non-Syllabus Project Report submitted in partial fulfilment of the requirements of

The award of the degree of

**Bachelor of Technology**

**in**

**CSE (Cyber Security)**

by

**Kushagra Vaishnav, Reg No: PCE23CY036**

**Nonit Bajaj, Reg No: PCE23CY046**

Under the guidance of

**Mr. Gaurav Sharma**

**Assistant Professor**

**Department of Advance Computing**



(Session 2024-25)

**Department of Advance Computing**

**Poornima College of Engineering**

ISI-6, RIICO Institutional Area, Sitapura, Jaipur – 302022

**Jan-June, 2025**

**DEPARTMENT CERTIFICATE**

This is to certify that **Kushagra Vaishnav,** registration no. **PCE23CY036**, **Nonit Bajaj** registration no. **PCE23CY046** of the IV semester Department of Advance Computing, has submitted this Project report entitled **Cloud-Based To-Do List Application with Serverless Backend** under the supervision of **Mr. Gaurav Sharma**, **Assistant Professor Department of Advance Computing,** working in division of Advance Computing as per the requirements of the Bachelor of Technology program at Poornima College of Engineering, Jaipur affiliated by Rajasthan Technical University.

| **Dr. Amol Saxena**  Head, Department of Advance Computing | **Mr. Gaurav Sharma**  NSP Guide |
| --- | --- |

**CANDIDATE’S DECLARATION**

We hereby declare that the work which is being presented in this project report entitled **Cloud-Based To-Do List Application with Serverless Backend** in the partial fulfilment for the award of the Degree of Bachelor of Technology in CSE(Cyber Security), submitted in the Department of Advance Computing, Poornima College of Engineering, Jaipur, is an authentic record of our work done during the period from **Jan 2025 to June 2025** under the supervision and guidance of **Mr. Gaurav Sharma, Assistant Professor, Department of Advance Computing**.

We have not submitted the matter embodied in this project report for the award of any other degree.

| Signature | Signature |
| --- | --- |
| Name of Candidate:  Registration no: | Name of Candidate:  Registration No.: |

Dated: 28/04/2025

Place: Jaipur

**SUPERVISOR’S CERTIFICATE**

This is to certify that, to the best of my knowledge, the candidate's above statement is correct.

| Dated:  Place: Jaipur | Mr. Gaurav Sharma  Assistant Professor  Department of Advance Computing |
| --- | --- |

**ACKNOWLEDGEMENT**

We would like to convey our profound sense of reverence and admiration to my supervisor, **Mr. Gaurav Sharma, Assistant Professor in the Department of Advance Computing at Poornima College of Engineering,** for her intense concern, attention, priceless direction, guidance, and encouragement throughout this research work.

We are grateful to **Dr. Mahesh Bundele**, **Principal & Director,** and **Dr. Pankaj Dhemla**, **Vice-Principal of Poornima College of Engineering,** for providing the necessary resources and a conducive environment to carry out this project.

Our special heartfelt gratitude goes to **Dr. Amol Saxena, HOD,** and **Dr. Kamlesh Gautam, Dy. HOD, Department of Advance Computing,** for unvarying support, guidance, and motivation during this project work.

We would like to express our deep sense of gratitude towards the management of Poornima College of Engineering, including **Shri Shashikant Singhi**, Chairman, Poornima Group, **Mr. M. K. M. Shah**, **Director General, Poornima Group,** and **Ar. Rahul Singhi, Director of Poornima Group,** for providing all the necessary resources and facilities required to complete this project.

We would like to take the opportunity to express our thanks to all faculty members of the Department for their kind support, technical guidance, and inspiration throughout the course.

We are also thankful to the non-teaching staff of the department for their support in the preparation of this dissertation work.

We are deeply thankful to my parents and all other family members for their blessings and inspiration. Last, but not least, we would like to give special thanks to God who enabled me to complete my dissertation on time.

**Kushagra Vaishnav, Department of** **Advance Computing, PCE23CY036**

**Nonit Bajaj, Department of** **Advance Computing, PCE23CY046**

**TABLE OF CONTENTS**

Contents

[***ABSTRACT***](#_32ts5idswiet) 1

[***Chapter 1: Introduction***](#_q5az3qaubyg8) 2

[***Chapter 2:  Literature Survey***](#_la13xk1hisym) 3

[***Chapter 3: Proposed Methodology***](#_6vevypdj54xk) 4

[***Chapter 4: Design and Implementation***](#_8qzxcm5zli8u) 5

[***Chapter 5: Experimental Results & Analysis:***](#_oll3gctrkiux) 6

[***Chapter 6: Conclusion and Future Scope***](#_fdcaxqg9p82p) 7

[***Referencing and Appendices***](#_1c13ek1bnfkf) 8

**LIST OF FIGURES**

| **S. No.** | **Fig. No.** | **Description** | **Page No.** |
| --- | --- | --- | --- |
| **1** | **1** | **Layout of the Frontend To-do list Application** | **5** |

**LIST OF ACRONYMS**

| **Serial Number** | **Acronym** | **Full Form** |
| --- | --- | --- |
| 1 | API | Application Programming Interface |
| 2 | CRUD | Create, Read, Update, Delete |
| 3 | UI | User Interface |
| 4 | UX | User Experience |
| 5 | AWS | Amazon Web Services |
| 6 | S3 | Simple Storage Service |
| 7 | CI/CD | Continuous Integration / Continuous Deployment |
| 8 | IDE | Integrated Development Environment |
| 9 | JSON | JavaScript Object Notation |
| 10 | DOM | Document Object Model |

# 

# ABSTRACT

The To-Do List application is designed to help users efficiently manage and track their tasks, improving productivity and organisation. Built using modern web technologies, the app features a user-friendly interface for adding, editing, deleting, and completing tasks. The application utilises a serverless architecture on AWS, leveraging services like AWS Lambda, Dynamodb, API Gateway, and S3 to ensure scalability, cost-efficiency, and minimal maintenance. With a responsive design, the app works seamlessly across various devices, including desktops and mobile phones. This project demonstrates the practical application of full-stack development and cloud-based technologies, offering users a reliable tool for personal task management.

**Keywords*:***

* To-Do List
* Task Management
* Productivity
* ReactJS
* AWS Lambda
* DynamoDB
* Serverless Architecture
* Cloud Computing
* Full-Stack Development
* API Gateway
* S3
* Responsive Design
* Web Application

**CHAPTER 1**

**INTRODUCTION**

* **Purpose of the Application:**
  + The To-Do List application is designed to help users manage daily tasks in an organised and efficient manner.
  + It ensures users can track their responsibilities, prioritise activities, and improve personal productivity.
* **Core Functionality:**
  + Users can **add**, **edit**, **delete**, and **mark tasks as complete**.
  + The interface is simple and intuitive, making it easy for users to manage tasks without any technical knowledge.
* **Technology Used:**
  + The frontend is developed using **React.js** for a responsive and dynamic user interface.
  + The backend is powered by **serverless architecture** using **AWS Lambda**, **Dynamodb** for data storage, **API Gateway** for routing, and **S3** for hosting static content.
* **Why Serverless Architecture:**
  + Serverless offers **scalability**, **cost-efficiency**, and **minimal maintenance**.
  + Resources scale automatically based on demand, making it ideal for lightweight applications like a To-Do List.
* **Cross-Platform Compatibility:**
  + The application is fully responsive, working smoothly across **desktop**, **tablet**, and **mobile** devices.

**CHAPTER 2**

**LITERATURE SURVEY**

1. **Task Management Applications:**
   * Existing tools like **Todoist**, **Microsoft To Do**, and **Google Tasks** offer task tracking, reminders, and collaboration.
   * These apps focus on productivity but often depend on centralised servers and paid tiers for full functionality.
2. **Cloud-Based Web Applications:**
   * Serverless architecture using **AWS Lambda**, **Dynamodb**, and **API Gateway** provides scalable and cost-efficient alternatives.
   * Literature shows that serverless apps reduce infrastructure management and auto-scale with user demand.
3. **Reactjs in Modern Web Development:**
   * React is widely used for building fast, interactive UIS.
   * Research highlights that its component-based structure improves reusability and maintainability.
4. **NoSQL Databases (DynamoDB):**
   * Nosql databases offer high availability and flexibility for unstructured data.
   * Suitable for applications with dynamic and real-time data requirements.
5. **Security and Authentication:**
   * Studies suggest integrating services like **AWS Cognito** enhances user authentication without managing backend identity systems.

**CHAPTER 3**

**PROPOSED METHODOLOGY**

**1. Requirement Analysis:**

* Identified core features: add, update, delete, and mark tasks as complete.

**2. Technology Stack Selection:**

* Choose **React.js** for the frontend.
* Used **AWS services** (Lambda, Dynamodb, S3, API Gateway) for backend and deployment.

**3. System Design:**

* Designed UI with a user-friendly interface.
* Created backend architecture using a serverless approach.

**4. Frontend Development:**

* Built task management components in React.
* Ensured responsive design for all devices.

**5. Backend Implementation:**

* Used Dynamodb to store task data.

**6. Integration:**

* Connected the frontend with the backend using API Gateway endpoints.

**7. Testing:**

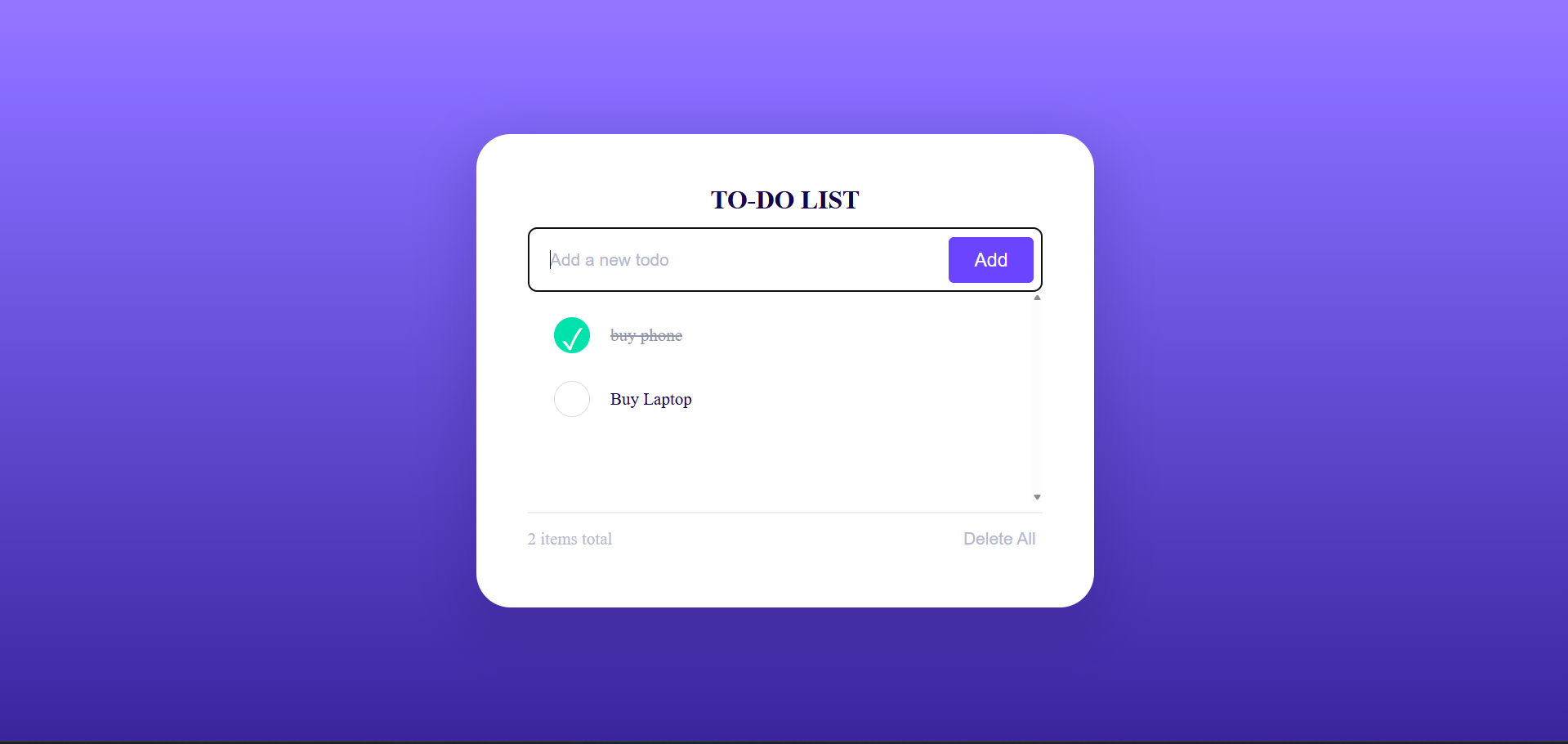
* Performed unit and integration testing.
* Ensured the app runs smoothly across browsers and devices.

**8. Deployment:**

* Hosted frontend on AWS S3 (static site).
* Configured backend using AWS services with proper roles and permissions.

## CHAPTER 4

## DESIGN AND IMPLEMENTATION



#### **Frontend (React.js)**

* **User Interface**: Developed using React.js, the frontend allows users to create, read, update, and delete tasks.
* **State Management**: Utilises React's useState and useEffect hooks for managing component state and side effects.
* **API Integration**: Communicates with the backend through HTTP requests to perform CRUD operations on tasks.

#### **Backend (AWS Serverless)**

* **AWS Lambda**: Handles the business logic for task operations (create, read, update, delete) in a serverless environment.
* **Amazon API Gateway**: Exposes RESTful endpoints that the frontend interacts with, routing requests to the appropriate Lambda functions.
* **Amazon Dynamodb**: The Nosql database stores task data with low latency and high availability.
* **AWS IAM**: Manages permissions and access control for AWS resources, ensuring secure service interactions.

#### **Deployment**

* **Frontend Hosting**: Deployed on Amazon S3, serving static files globally with low latency.

**CHAPTER 5**

**EXPERIMENTAL RESULTS & ANALYSIS**

* **Functionality Testing:**
  + **All core features (add, edit, delete, mark complete) worked as expected.**
  + **Task updates are reflected in real time using a cloud database (e.g., Dynamodb).**
  + **The application successfully handled multiple task entries without performance issues.**
* **Performance Analysis:**
  + **Fast response time due to serverless backend (AWS Lambda + API Gateway).**
  + **Minimal latency observed in CRUD operations.**
  + **Static content hosted on S3 loads quickly and consistently.**
* **Scalability Testing:**
  + **The application scaled well under multiple user simulations (handled by AWS auto-scaling).**
  + **No downtime during deployment or usage.**
* **Cross-Platform Testing:**
  + **Web app rendered properly on various browsers (Chrome, Firefox, Edge).**
  + **Responsive design works on desktop and mobile devices.**
* **Error Handling:**
  + **Proper alerts and validations are implemented for empty tasks and invalid actions.**
  + **Backend errors are logged for debugging using AWS CloudWatch.**

**CHAPTER 6**

**CONCLUSION AND FUTURE SCOPE**

### Conclusion

* The To-Do List app helps users manage tasks efficiently.
* Core features include add, edit, delete, and mark tasks as complete.
* Built using serverless architecture (e.g., AWS Lambda, Dynamodb, S3).
* Demonstrates cloud-based app deployment and scalability.
* Strengthened skills in full-stack and cloud development.

### Future Scope

* Add user authentication (e.g., AWS Cognito).
* Integrate task reminders via email or push notifications.
* Develop a mobile version using React Native or Flutter.
* Enable task sharing and team collaboration.
* Include analytics for productivity tracking.
* Support voice commands (e.g., Alexa, Google Assistant).

**REFERENCES**

### References

1. **Amazon Web Services Documentation** [**https://docs.aws.amazon.com**](https://docs.aws.amazon.com)
2. **AWS Lambda – Serverless Compute** [**https://aws.amazon.com/lambda**](https://aws.amazon.com/lambda)
3. **Amazon DynamoDB – NoSQL Database** [**https://aws.amazon.com/dynamodb**](https://aws.amazon.com/dynamodb)
4. **Amazon S3 – Cloud Storage** [**https://aws.amazon.com/s3**](https://aws.amazon.com/s3)
5. **AWS API Gateway – API Management** [**https://aws.amazon.com/api-gateway**](https://aws.amazon.com/api-gateway)
6. **Reactjs Official Documentation** [**https://reactjs.org/docs/getting-started.html**](https://reactjs.org/docs/getting-started.html)
7. **GitHub – Project Repository (if applicable)**[**https://github.com/pmuens/serverless-book/tree/master**](https://github.com/pmuens/serverless-book/tree/master)