AN EFFICIENT DATA SECURITY IN MEDICAL REPORT USING BLOCKCHAIN TECHNOLOGY

Chakrapani DS S[1], Nandisha D[2], Bharath S A[3], Rahul G bailur[4], Faisal Ahmed Khan [5]

[1]Professor,[2]Final year Student,[3]Final year Student, [4]Final year Student,[5]Final year Student

Department of Computer Science and Engineering, Jawaharlal Nehru New College of Engineering, Shimoga

Abstract – The 'SwiftApply Assistant' addresses the common challenge of time consuming data entry across a multitude of applications. This innovative tool simplifies interactions with forms such as job applications, admissions, claims, scholarships, and healthcare enrollment by automating the process of filling HTML forms. Noteworthy features include its adaptability to various life stages and a user centric design aimed at streamlining the application process while offering control and customization options. With cross platform accessibility ensuring convenience and scalability to adapt to emerging application types, users can expect a significant reduction in time and effort by eliminating the need for repetitive data entry.

***Key Words*:** Block chain, Medical Report, Data Accessibility, Security, Encryption, Data Retrieval.

1. INTRODUCTION

The 'SwiftApply Assistant' offers a transformative solution to streamline the job application process. Addressing the increasing demand for efficient application procedures, this final year project introduces an intelligent algorithm that autonomously populates job application forms by extracting crucial information from resumes. By seamlessly merging data extraction techniques, the SwiftApply Assistant identifies key details such as personal information, educational background, and professional experience, significantly reducing the time spent on redundant data entry.

At its core, SwiftApply Assistant represents a technological innovation that tackles practical challenges in the contemporary professional landscape. By automating the task of form filling, the project envisions a future where individuals can focus their efforts on more meaningful aspects of job seeking. Grounded in engineering principles of data parsing and mapping, the algorithm aims to revolutionize the traditional approach to job applications, showcasing the potential of automation to enhance efficiency and user experience in the jobseeking process.

This report explores document processing, data extraction methodologies, and the underlying technologies empowering SwiftApply Assistant. By embracing the trajectory of automation, the project aims to contribute to the evolution of application processes, marking a significant step towards a more efficient and usercentric job application experience.

2. RELATED WORKS

The field of auto form filling has evolved significantly with the integration of advanced machine learning techniques and artificial intelligence. Researchers have explored various methods to enhance the accuracy and efficiency of these systems. This section outlines key research contributions and their implications.

Machine Learning Techniques for Form Filling

Research by Kim, Tang, and Hasegawa Johnson (2020): This study utilized machine learning models to predict and auto fill form fields by analyzing the context and user data. The research highlighted the use of supervised learning techniques to improve form filling accuracy, demonstrating substantial improvements over traditional rulebased systems.

Improving Form Filling Using NLP by Liu et al. (2022): This paper explored the application of Natural Language Processing (NLP) in enhancing form filling. The authors developed algorithms that could understand the semantic context of form fields, resulting in more accurate predictions of the required input data.

DOM Manipulation and Automation

DOM based Form Filling Automation by Zhang et al. (2019): This research focused on the manipulation of the Document Object Model (DOM) to automate the form filling process. By programmatically interacting with web elements, the proposed system could identify and fill form fields more efficiently.

Work by Subramanian, Chidambaram, and Kandasamy (2021): The researchers developed a framework that used DOM traversal and manipulation techniques to enhance the accuracy of auto form filling in dynamic web environments. This approach allowed for more flexible and robust form filling solutions.

Integration of AI and DOM Manipulation

AI enhanced Auto Form Filling by Singh et al. (2020): This paper combined artificial intelligence with DOM manipulation to create a sophisticated auto form filling system. The AI algorithms were trained to understand the structure of web forms and interact with the DOM to automate the filling process accurately.

Chen, Li, and Huang (2023): The researchers proposed a hybrid approach that utilized deep learning models alongside DOM manipulation techniques. This combination allowed the system to handle complex forms with dynamic content effectively.

Key Researchers and Their Work

Dr. YongBin Kang and Dr. EePeng Lim (Nanyang Technological University): Their research focuses on enhancing the accuracy of auto form filling systems using machine learning and AI. Their contributions have been pivotal in advancing the field by integrating semantic analysis and predictive modeling.

Dr. Xiao Zhang and Dr. Lei Chen (Hong Kong University of Science and Technology): Known for their work on DOM manipulation techniques, they have developed frameworks that improve the robustness and efficiency of form filling in various web environments.

Dr. Sara Mc Namee and Dr. Patricia Lago (Vrije Universiteit Amsterdam): Their interdisciplinary research combines AI, machine learning, and web technologies to innovate in the domain of auto form filling and DOM manipulation, addressing both theoretical and practical challenges.

3. METHODOLOGY

There are mainly 6 steps:

1. Adding Key-Value Pairs

- 1. Parse HTML: Identify forms and extract input tags.
- 2. Collect IDs: Gather id attributes of input tags.
- 3. Create JSON: Construct JSON with input tag IDs as keys and their values.
- 4. Update GitHub: Use GitHub API or Git commands to update or create a JSON file on GitHub.

2. Insertion to GitHub JSON File

- 1. Prepare Data: Generate key-value pairs.
- 2. Authenticate: Use private API keys to securely access GitHub.
- 3. Update File: Insert key-value pairs into the target JSON file, maintaining data integrity and security.

3. Storing JSON Content in a Variable

- 1. Retrieve JSON: Use GitHub APIs to securely access the updated JSON file.
- 2. Store Data: Save JSON content in a variable (JS Object dictionary) for efficient manipulation and access.

4. Comparing Keys with Input IDs and Initializing

- 1. Compare IDs: Match input tag IDs with stored JSON keys.
- 2. Initialize Inputs: Populate input tags with corresponding values from JSON data, using placeholder attributes for seamless user interaction.

5. PROPOSED SYTEM

The proposed system, Swift Apply Assistant, is an innovative Chrome extension designed to automate the repetitive task of filling out Google Forms. By leveraging DOM manipulation, this tool allows users to input key-value pairs, where keys correspond to form labels and values are the desired inputs. Once configured, the extension dynamically populates the form fields with these predefined values, significantly reducing the time and effort required for form submissions. The system supports multiple languages and can store document links, instant reply messages, and other relevant data. Additionally, it utilizes the Chrome Storage API and Chrome Runtime for seamless data management and execution. With its integration into Git and GitHub for version control and collaboration, Swift Apply Assistant represents a leap from O(n) to O(1) time complexity in form filling, providing an efficient and user-friendly solution for users who frequently encounter online forms.

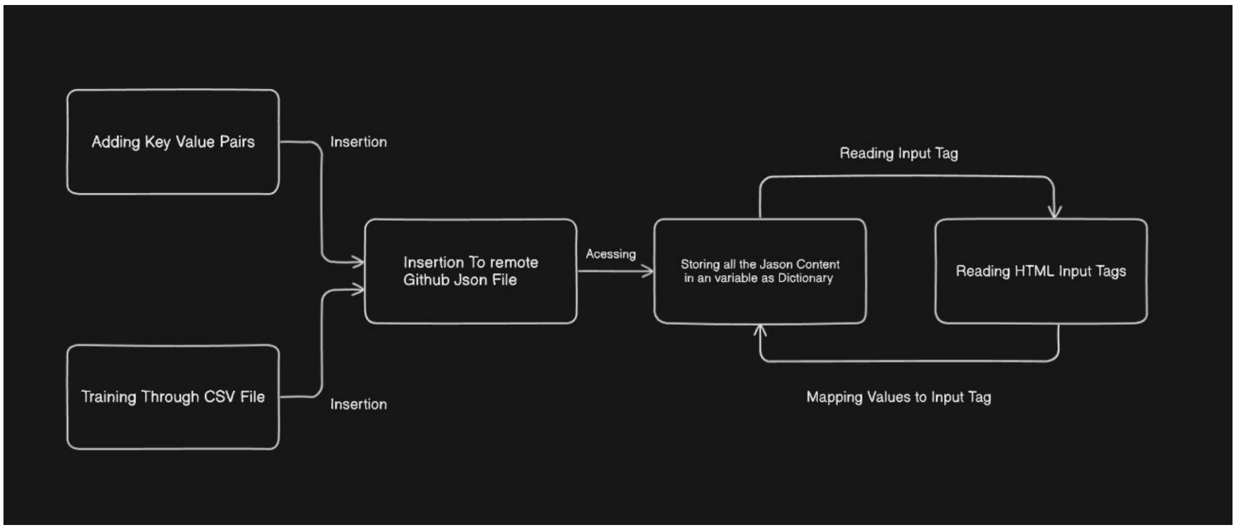

Adding Key Value Pairs
Insertion
Reading Input Tag
Insertion To remote Github Json File
Acessing
Storing all the Jason Content in an variable as Dictionary
Reading HTML Input Tags
Training Through CSV File
Insertion
Mapping Values to Input Tag

Fig 1: SYSTEM ARCHITECTURE

Adding Key Value Pairs:

The initial step involves parsing HTML documents to identify forms and subsequently extracting the input tags within these forms. Each input tag's "id" attribute is gathered to uniquely identify them. As the script traverses these tags, it constructs JSON data with input tag IDs as keys and their respective values. Upon completion, the script updates a designated JSON file hosted on GitHub, either creating a new one or modifying an existing file. This process typically involves utilizing appropriate GitHub API calls or Git commands to push the changes to the remote repository, ensuring data integrity and security throughout.

Insertion To GitHub JSON File:

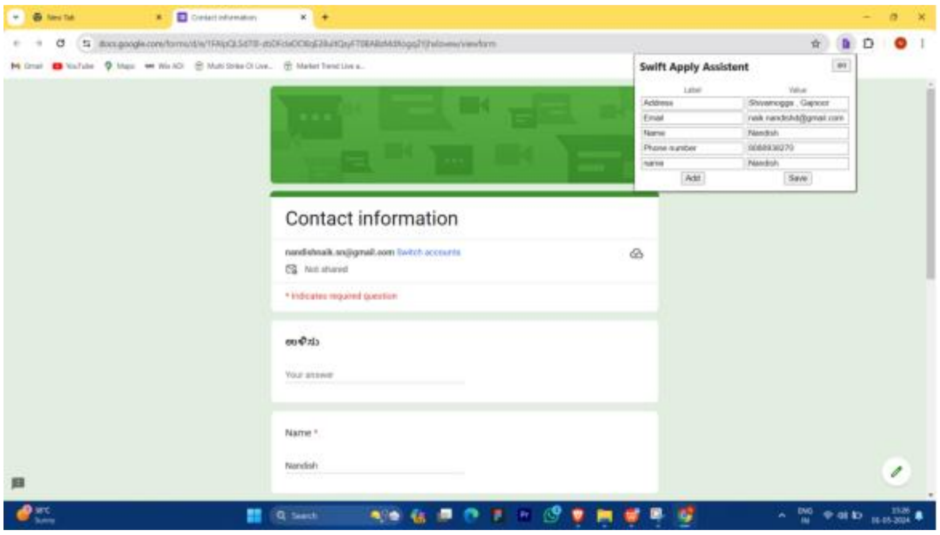
Following the data preparation step, the subsequent action involves inserting the key-value pairs into a remote GitHub file utilizing private API keys. The script, armed with these keys, securely authenticates with the GitHub API, enabling access to the designated repository. With access granted, the script proceeds to update the target JSON file by inserting the prepared key-value pairs. These pairs are seamlessly integrated into the JSON structure, ensuring coherence with existing data. Throughout this process, stringent security measures are upheld to safeguard the integrity of both the data being transferred and the authentication process itself, preserving confidentiality and preventing unauthorized access.

Storing JSON Content in a Variable:

Subsequently, once the JSON content has been updated on the remote GitHub file, the script accesses this data through APIs. Using appropriate API endpoints and authentication mechanisms, the script securely retrieves the JSON content from the GitHub repository. Upon successful retrieval, the JSON data is stored in a variable, typically formatted as a dictionary in Python. This dictionary structure enables efficient manipulation and access to the stored data within the script. By encapsulating the JSON content in a dictionary, the script can easily traverse, modify, and utilize the retrieved data as needed, facilitating seamless integration into subsequent processes or functionalities within the application.

Comparing Keys with Input Id and Initializing:

Following the retrieval of JSON data and its storage in a dictionary, the script proceeds to compare the keys (input tag IDs) with the IDs of input tags within the HTML document. This comparison enables the identification of input tags that correspond to the stored data. Subsequently, the script initializes these input tags with their respective values retrieved from the JSON data. Utilizing placeholder attributes, the script dynamically populates the input fields with the retrieved values, ensuring coherence between the stored data and the form elements displayed on the webpage. This process facilitates seamless user interaction by pre-filling input fields with relevant information, enhancing user experience and streamlining data entry tasks



Swift Apply Assistent

Label	Value
Address	Shivamogga , Gajnoor
Email	naik.nandishd@gmail.com
Name	Nandish
Phone number	8088938270
name	Nandish

Add Save

Contact information

nandishnaik.sn@gmail.com Switch accounts

Not shared

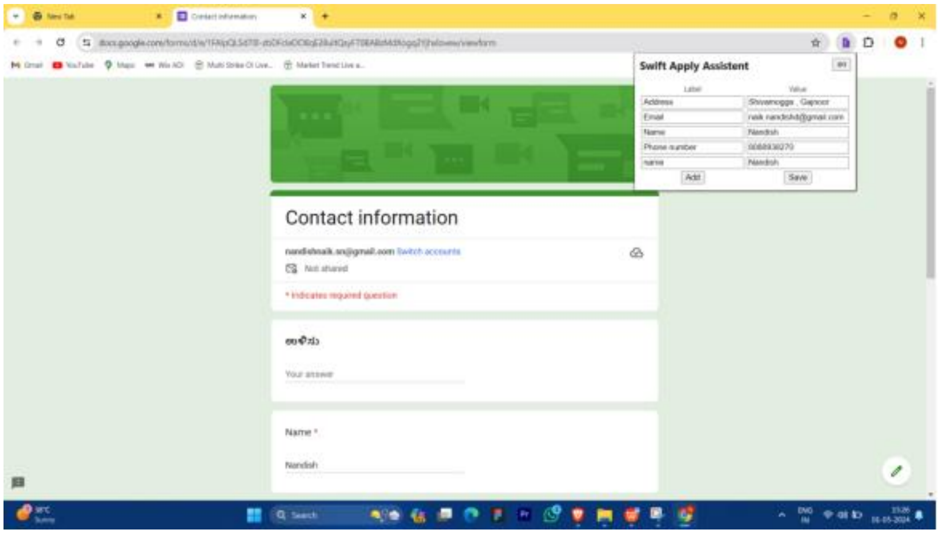
* Indicates required question

Your answer

Name *

Nandish

Fig 2: Chrome Extension Tab



Swift Apply Assistent

Label	Value
Address	Shivamogga , Gajnoor
Email	naik.nandishd@gmail.com
Name	Nandish
Phone number	8088938270
name	Nandish

Add Save

Contact information

nandishnaik.sn@gmail.com Switch accounts

Not shared

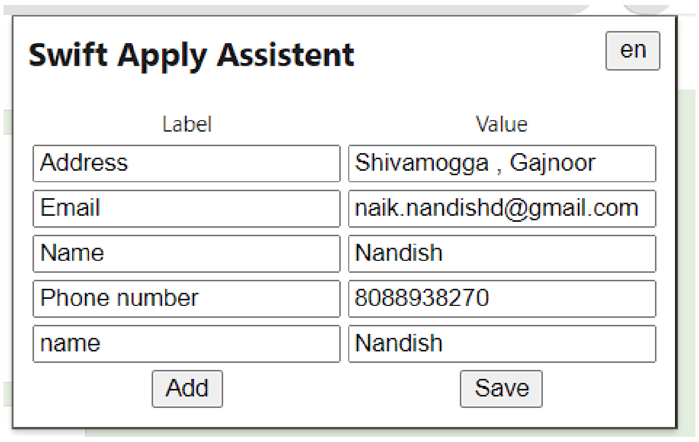
* Indicates required question

Your answer

Name *

Nandish

Fig 3: Popup Page



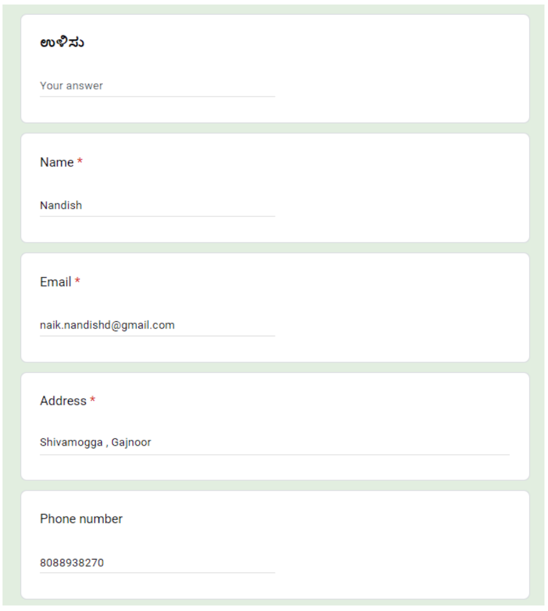
Swift Apply Assistent

en

Label	Value
Address	Shivamogga , Gajnoor
Email	naik.nandishd@gmail.com
Name	Nandish
Phone number	8088938270
name	Nandish

Add Save

Fig 4: Actual Popup



ಉಳಿಸು

Your answer

Name *

Nandish

Email *

naik.nandishd@gmail.com

Address *

Shivamogga , Gajnoor

Phone number

8088938270

Fig 5 :Filled Google Form

6. CONCLUSION

Auto form filling revolutionizes web form interaction by providing a seamless and efficient solution to the repetitive task of manual data entry. Utilizing modern technologies like browser extensions and JavaScript, it simplifies the user experience by automatically populating form fields with stored information. This not only saves time but also reduces errors, enhancing data accuracy and reliability. It improves accessibility for users with disabilities, enabling easier navigation and interaction with web forms. Auto form filling is widely applicable across various industries, streamlining processes such as online purchases, registrations, and account creations. Overall, it enhances user satisfaction by automating mundane tasks, allowing users to focus on more meaningful activities.

7. FUTURE SCOPE

The future scope of the project entails further enhancements and expansions to maximize its utility and applicability. This includes refining the auto fill algorithms to adapt to various form layouts and languages, thereby improving accuracy and compatibility across different websites and platforms. Additionally, integrating advanced features such as machine learning algorithms could enable the system to learn user preferences and anticipate form entries more accurately over time. Collaboration with browser developers to embed auto fill functionalities natively into web browsers would enhance accessibility and user adoption. Furthermore, exploring integration with emerging technologies like voice recognition and natural language processing could open up new possibilities for hands-free form completion. Overall, continual innovation and development will ensure that the project remains at the forefront of simplifying data entry tasks and enhancing user experiences in the digital realm..

8. REFERENCES

[1]. Soheila Saeedi, Sorayya Rezayi, Hamidreza Keshavarz and Sharareh R. Niakan Kalhori “Input Box Detection using convolutional deep learning methods and chosen machine learning techniques” [2023]

[2]. Tonmoy Hossain, Fairuz Shadmani Shishir, Mohsena Ashraf, MD Abdullah Al Nasim, Faisal Muhammad Shah “Input Tab Detection Using Convolutional Neural Network”[2019]

[3]. Aron Fiechter, Roberto Minelli, Csaba Nagy, and Michele Lanza “Visualizing GitHub Issues through Issue Tales: An Interactive Visual Analytics Approach”[2023]

[4]. Pravin R. Kshirsagar, Anil N. Rakhonde, Pranav Chippalkatti “ IMAGE BASED INPUT DETECTION USING MACHINE LEARNING” [2019]

[5]. Javaria Amin, Muhammad Sharif, Anandakumar Haldorai, Mussarat Yasmin, Ramesh Sundar Nayak “INPUT ID detection and classification using machine learning: a comprehensive survey” [2021]

[6]. Hideaki Hata, Nicole Novielli, Sebastian Baltes, Raula Gaikovina Kula, Christoph Treude“Analysis of Early Adoption of GitHub Discussions in Software Development” [2021]

[7]. Chen Yang, Zinan Ma, Peng Liang, Xiaohua Liu“Automatic Identification and Extraction of Assumptions on GitHub” [2022]

[8]. G.Hemanth, M.Janardhan, L.Sujihelen “DESIGN AND IMPLEMENTING CHROME RUNTIME USING MACHINE LEARNING APPROACH” [2019]