|  |  |
| --- | --- |
| **Project Name :** | **Academic Year :** |
| **Subject Name:** | **Semester : Sixth** |

**A STYDY ON**

Study on **Topic Name**

***MICRO PROJECT REPORT***

**Submitted in March 2025 by the group of……3….students**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No** | **Roll No (Sem-vi)** | **Full name of Student** | **Enrollment No** | **Seat No (Sem-vi)** |
| 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |

Under the Guidance of

[ your guide name ]

in

Three Years Diploma Program in Engineering & Technology of Maharashtra State Board of Technical Education, Mumbai (Autonomous)

ISO 9001:2008 (ISO/IEC-27001:2013)

at

[ your college name ]



**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION, MUMBAI**

***Certificate***

This is to certify that Mr. /Mrs.

Roll No: of **Sixth Semester**  of **Diploma in Engineering & Technology** at **[ your college name ] ,** has completed the **Micro Project** satisfactorily in Subject **Subject Name** in the academic year 2024-2025 as per the MSBTE prescribed curriculum of I Scheme.

Place: Pune Enrollment No:

Date: / / 2025 Exam Seat No:

# Project Guide Head of the Department Principal

Head of Institute

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr.** | **Title** | **Page No .** |
|  | Abstract |  |
| 1. | Introduction |  |
| 2. | Literature Survey |  |
| 3. | System Design |  |
| 4. | Data Flow Diagram |  |
| 5. | Use Case Diagram |  |
| 6. | Testing and Evaluation |  |
| 7. | Conclusions |  |
| 8. | References |  |

**Abstract**

In the modern education system, teachers often struggle with manually creating assignments, which can be time-consuming and inefficient. To address this issue, the **Assignment Maker for Teachers** is developed as a **web-based tool** that automates the process of generating assignments. This project allows teachers to **input essential details** such as **college name, teacher name, subject, and submission date** and specify the number of questions required. Based on this input, the system dynamically generates a form where teachers can select questions from a predefined dropdown list. Once finalized, the assignment is converted into a **professionally formatted PDF** with structured sections, including **bold headings, dividers, and a watermark of the college name** for authentication.

This project is implemented using **HTML, CSS, JavaScript, PHP, and FPDF library**, ensuring an intuitive user experience and smooth PDF generation. The **primary objective** of this system is to **minimize manual efforts, enhance productivity, and provide a standardized format for assignment creation**. Additionally, it supports **large assignments** (more than 20 questions) by dynamically adjusting content layout while maintaining a fixed professional appearance.

By automating the assignment-making process, this project significantly **reduces the workload for educators**, enabling them to focus more on teaching rather than administrative tasks. This tool serves as a valuable asset for educational institutions by **streamlining assignment creation and ensuring consistency in document formatting**.

**Introduction**

#### ****1. Background and Motivation****

Education is one of the most critical sectors that continuously evolves with the advancement of technology. In modern classrooms, assignments play a vital role in assessing students' knowledge, improving their understanding, and encouraging independent learning. However, for teachers, manually preparing assignments can be a tedious and time-consuming task, especially when dealing with multiple subjects and large student batches. The need for an efficient system that **automates assignment creation** has become essential in educational institutions.

To address this issue, the **Assignment Maker for Teachers** is designed as a **web-based tool** that simplifies the process of generating assignments. The primary goal of this project is to provide **a user-friendly platform** where teachers can quickly create structured assignments by selecting questions from a predefined list and generating a **professionally formatted PDF**. This tool ensures a standardized layout, eliminates manual formatting errors, and significantly reduces the time spent on preparing assignments.

#### ****2. Problem Statement****

Traditional methods of assignment creation involve **manual selection, typing, formatting, and printing**, which is not only time-consuming but also prone to inconsistencies. Teachers often have to create assignments repeatedly for different subjects, requiring them to format each document from scratch. Additionally, formatting errors such as misalignment, excessive spacing, or inconsistent fonts can make assignments look unprofessional.

Key problems faced by educators in manual assignment creation:

* **Time-consuming process** of manually formatting and structuring assignments.
* **Inconsistencies in layout and formatting** due to human errors.
* **Difficulty in reusing questions** from previous assignments.
* **Lack of automation** for organizing and generating assignment PDFs.

This project aims to solve these issues by providing an **automated** and **structured** way to create assignments, ensuring efficiency and uniformity in document presentation.

#### ****3. Objectives of the Project****

The **Assignment Maker for Teachers** is developed to:

1. **Automate the assignment creation process** by allowing teachers to input details and dynamically generate questions.
2. **Provide a structured PDF output** with sections for college name, teacher name, subject,

**Literature Survey**

#### ****1. Introduction to Assignment Generation Systems****

In the traditional education system, teachers manually create assignments, which involves selecting relevant questions, formatting the document, and distributing it to students. This process is time-consuming, error-prone, and lacks standardization. With the advancements in **web-based automation**, various digital solutions have been introduced to assist educators in generating assignments efficiently.

The **Assignment Maker for Teachers** aims to automate this process by allowing teachers to **select questions dynamically, input assignment details, and generate a professionally formatted PDF**. To develop this project, a study of existing assignment generation tools, document formatting technologies, and PDF generation techniques was conducted.

#### ****2. Existing Systems and Their Limitations****

Several digital platforms and learning management systems (LMS) such as **Google Classroom, Moodle, and Blackboard** provide assignment creation features. However, these systems have certain limitations:

* ❌ **Complexity** – Many LMS platforms require extensive configuration, making it difficult for teachers to quickly generate assignments.
* ❌ **Limited Customization** – These systems often lack the ability to dynamically format the document with **bold headings, dividers, and watermarks**.
* ❌ **Internet Dependency** – Some online tools require a continuous internet connection, which can be a barrier for users with limited access.
* ❌ **Lack of Automated Formatting** – Most traditional systems do not provide **automated PDF formatting with fixed margins and section-wise content structuring**.

Given these challenges, a **lightweight, web-based** solution is necessary to allow teachers to **easily create assignments with professional formatting in just a few clicks**.

#### ****3. Study of PDF Generation Technologies****

The **FPDF (Free PDF) library** was selected for generating assignment PDFs because of the following advantages:

✔ **Lightweight and Fast** – Requires minimal server resources.  
✔ **Supports Custom Formatting** – Allows adding **bold headings, tables, dividers, and watermarks**.  
✔ **Dynamic Content Handling** – Automatically adjusts layout when the number of questions increases.  
✔ **Offline Access** – Once generated, the assignment PDF can be **downloaded and shared** without requiring an internet connection.

Other alternatives like **TCPDF and DOMPDF** were considered but found to be heavier and slower for simple document generation tasks.

#### ****4. Web Technologies for User Interface Development****

To develop an **interactive and user-friendly** assignment generator, the project uses:

* **HTML & CSS** – For designing an **intuitive, well-structured form**.
* **JavaScript** – For dynamically displaying input fields based on the **number of questions** entered.
* **PHP** – For handling form submissions and integrating with the **FPDF library** to generate PDFs.

**System Design**

#### 1. ****Overview****

The Automated Assignment Maker is a web-based application designed to help teachers create assignments quickly and efficiently. It allows teachers to input college name, teacher name, subject, submission date, and the number of questions they want to include in an assignment. Based on the entered number, predefined questions are dynamically displayed. The user can select these questions, and once completed, the application generates a professional-looking PDF with all details and questions included.

#### 2. ****System Architecture****

The system follows a **client-server architecture** with the client (front-end) interacting with the server (back-end). It is a dynamic web application where the client interacts with the web page built using **HTML, CSS, and JavaScript**, and the server side uses **PHP** to handle form data, generate dynamic content, and create the final PDF.

* **Client-side (Frontend)**: The client interacts with the system through a user interface (UI) created with HTML and CSS for layout, JavaScript for interactivity, and PHP for server-side processing.
* **Server-side (Backend)**: The PHP scripts process the input, handle dynamic question generation, and use libraries like **TCPDF** or **FPDF** to generate the PDF document.
* **Database**: While not mandatory, a database (e.g., MySQL) can be added for storing predefined questions, templates, and records of generated assignments.

#### 3. ****Modules and Components****

##### 3.1 ****Input Form Module (Frontend)****

* **College and Teacher Details**:
  + Input fields for College Name, Teacher Name, Subject, and Submission Date.
  + These fields are essential for personalizing the assignment.
* **Dynamic Question Generator**:
  + Input field to specify the number of questions to be added.
  + A JavaScript function dynamically adds question fields, each labeled with a number (e.g., Question 1, Question 2, etc.).
  + Each question will have a dropdown to select from predefined questions stored in the system or manually entered.
* **Submit Button**:
  + When the user clicks the "Generate Assignment" button, the data entered in the form is sent to the PHP server for processing.

##### 3.2 ****Backend Processing (PHP)****

* **Data Handling**:
  + Collect the input data (college name, teacher name, subject, etc.) from the submitted form.
  + Process the data to generate the structure of the assignment.
* **Question Selection Logic**:
  + For each dynamic question field, a selection is made based on the user’s choice from the dropdown (either predefined or manually entered).
  + Predefined questions could be fetched from a **MySQL database** (if implemented) or stored in an array for a simpler approach.
* **PDF Generation**:
  + Use **TCPDF** or **FPDF** to generate the PDF document. The generated document should include:
    - **College, Subject, Teacher Details** in bold at the top.
    - A **line separator** between the details and questions.
    - **Watermark** of the college name.
    - The questions listed in a clear and readable format.

##### 3.3 ****PDF Layout****:

* **Section 1**: College, Teacher, and Subject details, formatted in bold, followed by a divider line.
* **Section 2**: List of selected questions, with clear numbering and formatting.
* **Watermark**: The college name in a transparent, light font across the background of the PDF.
* **Fixed top margin** to ensure that the content starts 100px from the top, regardless of the number of questions.

##### 3.4 ****Optional: Database Integration****

* **MySQL Database** (optional):
  + A table to store **predefined questions** that teachers can select when creating an assignment.
  + Table structure might include columns like: id, question\_text, subject, difficulty\_level, etc.
* **Storing Assignment Details** (optional):
  + Another table can be used to store the generated assignments for later use, which could include details like teacher name, subject, number of questions, and PDF file path.

##### 3.5 ****PDF Generation Library****

* **TCPDF** or **FPDF**: These PHP libraries can be used to generate the PDF dynamically. Both libraries offer the ability to format text, add images (e.g., a watermark), draw lines (e.g., for separators), and handle complex layouts.

#### 4. ****Flow Diagram****

1. **Teacher inputs assignment details** (college, teacher, subject, submission date) and the number of questions.
2. **Dynamic question fields** appear based on the input number.
3. Teacher selects predefined questions or writes new ones.
4. **Form submission** sends data to the server.
5. **PHP backend processes data** and generates a PDF using TCPDF or FPDF.
6. **PDF is generated** with formatted content, watermarks, and separators.
7. The teacher **downloads the generated PDF**.

#### 5. ****Technologies Used****

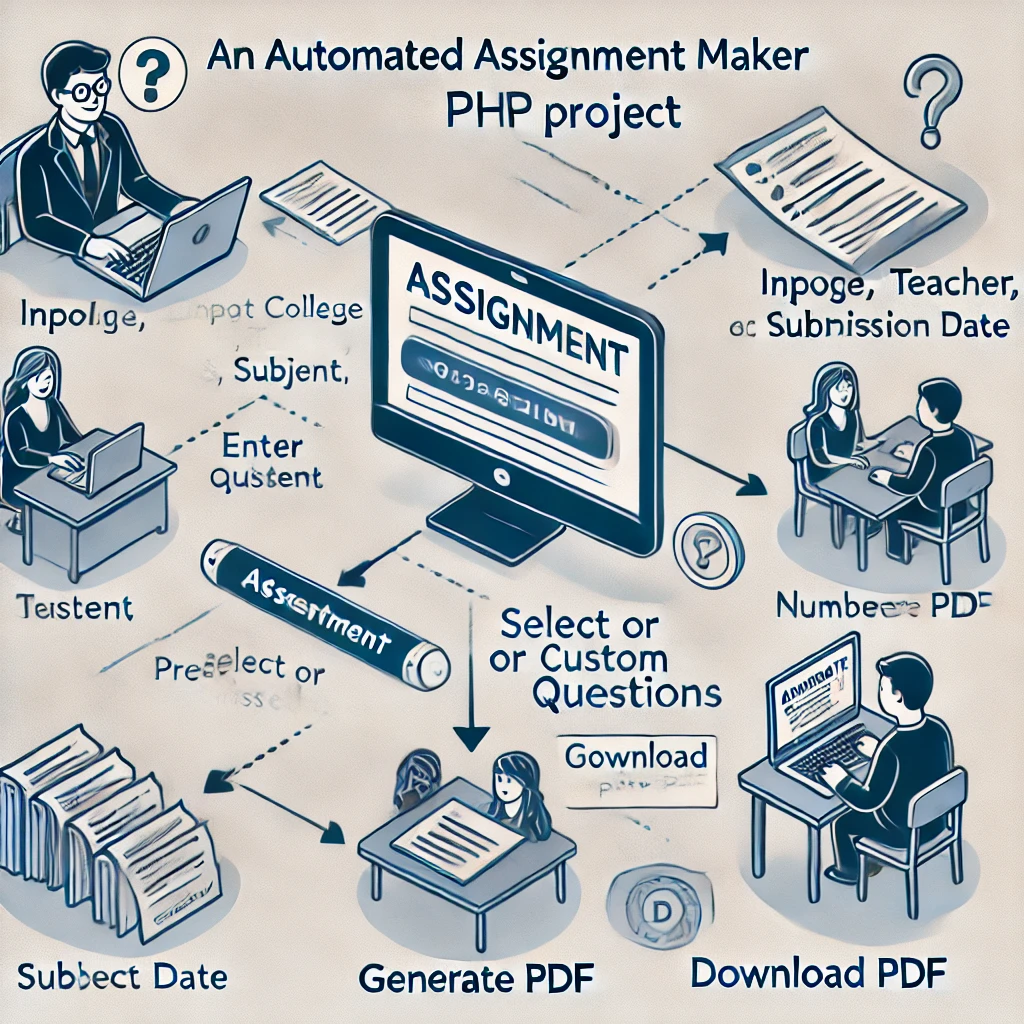
* **Frontend**: HTML, CSS, JavaScript
* **Backend**: PHP
* **PDF Generation**: TCPDF or FPDF
* **Database** (Optional): MySQL (for predefined questions and storing generated assignments)

#### 6. ****Security Considerations****

* **Data Validation**: Ensure proper validation of all user inputs (e.g., college name, teacher name) to prevent malicious input.
* **Sanitization**: All user-provided data should be sanitized to avoid **XSS** (Cross-Site Scripting) attacks.
* **File Handling**: If the application stores generated PDFs, ensure secure handling of uploaded and stored files.

**Data Flow diagram**

**Use case Diagram**



**Testing and Evaluation**

#### 1. ****Introduction to Testing****

Testing is a crucial part of the software development life cycle that ensures the application works as intended. In this project, testing will be done to verify that the **Automated Assignment Maker** system is functioning correctly, both from a **functional** and **non-functional** perspective.

#### 2. ****Types of Testing****

##### 2.1 ****Unit Testing****

Unit testing will be performed to test individual components of the system. This involves testing individual PHP functions, JavaScript functions, and other isolated pieces of logic.

**PHP Functions**:

* + **Input Validation**: Ensure that inputs like college name, teacher name, subject, etc., are correctly validated.
  + **PDF Generation Logic**: Test that the generated PDF includes all required sections (college details, subject, teacher details, questions).
  + **Watermarking**: Verify that the watermark (college name) is correctly applied to the generated PDF.

**JavaScript Functions**:

* + **Dynamic Question Fields**: Test that when the user specifies the number of questions, the correct number of question fields are generated.
  + **Dropdown Selection**: Ensure the dropdown options correctly display predefined questions or allow custom questions to be entered.

##### 2.2 ****Integration Testing****

Integration testing focuses on verifying the interaction between different modules and ensures that they work together seamlessly.

* **Form Submission**: Test the flow of data from the input form (Frontend) to the PHP server (Backend). Ensure that the form data is correctly submitted and processed.
* **Question Selection and Display**: Test if the selected predefined questions or custom inputs are displayed correctly on the form and passed to the backend for PDF generation.
* **PDF Generation**: Ensure that once the form is submitted, the PHP server correctly generates the PDF, including all details, questions, watermark, and formatting.

##### 2.3 ****System Testing****

System testing involves testing the entire system as a whole to ensure that all components and modules work together correctly.

* **End-to-End Test**:
  + Input the college name, teacher name, subject, and submission date into the form.
  + Specify a number of questions (e.g., 5) and select predefined questions or enter custom ones.
  + Submit the form and verify that a PDF is generated with the correct content (college name, subject, teacher, questions, watermark, and formatting).
  + Verify that the PDF can be downloaded successfully and opened on various PDF readers.

##### 2.4 ****User Acceptance Testing (UAT)****

This type of testing ensures the system meets the requirements of the intended users, in this case, the teachers. Teachers will test the system by:

* Creating assignments with various combinations of questions and details.
* Checking if the generated PDFs meet their expectations (professional look, watermark, correct format).
* Providing feedback regarding ease of use, layout, and overall experience.

##### 2.5 ****Performance Testing****

Performance testing ensures that the system performs well under different loads.

* **Load Testing**:
  + Simulate multiple users accessing the application at the same time to ensure the system can handle the traffic without slowing down or crashing.
* **Response Time**:
  + Test the time taken to generate the PDF after form submission to ensure that the system responds within an acceptable time (less than 5 seconds for PDF generation, for example).

##### 2.6 ****Security Testing****

Security testing ensures that the application is secure and free from vulnerabilities.

* **SQL Injection**: Test form inputs for potential vulnerabilities to SQL injection attacks.
* **Cross-Site Scripting (XSS)**: Ensure that no malicious scripts can be injected through form fields or URLs.
* **File Upload Security**: If files are uploaded (e.g., for predefined questions), ensure that the system checks for harmful files and prevents any security risks.
* **Session Management**: Ensure that user sessions are securely managed, and that session timeouts and authentication processes work properly.

##### 2.7 ****Compatibility Testing****

This ensures that the application works across various environments.

* **Cross-Browser Testing**:
  + Ensure that the application functions properly on all major browsers (Chrome, Firefox, Safari, Edge) with consistent layout and functionality.
* **Cross-Device Testing**:
  + Test the application on different devices, such as desktops, laptops, and mobile phones, ensuring that the layout remains responsive.

#### 3. ****Evaluation Criteria****

The evaluation of the system will be based on the following criteria:

##### 3.1 ****Functional Evaluation****

* **Correctness of Data Processing**: Ensure that all input data is correctly captured, processed, and displayed in the generated PDF.
* **PDF Generation**: Verify that the PDF output meets the specified design and formatting requirements (e.g., bolded text, watermarks, dividers, and proper question listing).

##### 3.2 ****Usability Evaluation****

* **User Interface (UI)**: Evaluate the ease of use of the input form and whether teachers can easily navigate through the assignment creation process.
* **Accessibility**: Check if the form is user-friendly, accessible, and intuitive.

##### 3.3 ****Performance Evaluation****

* **Speed**: Evaluate how quickly the system generates PDFs, especially under varying loads.
* **Scalability**: Test whether the system remains responsive and efficient when handling a higher number of questions or users.

##### 3.4 ****Security Evaluation****

* **Data Protection**: Ensure that user data (e.g., college, teacher details) is properly protected and that no unauthorized access occurs.
* **Protection Against Attacks**: Evaluate the system’s resilience against common web attacks (SQL injection, XSS, etc.).

#### 4. ****Test Cases****

Here are a few sample test cases that could be used to evaluate the system:

* **Test Case 1**: Verify that when the number of questions is set to 3, exactly 3 question fields are dynamically generated.
* **Test Case 2**: Submit the form with valid college and teacher details and verify that the generated PDF contains these details correctly formatted and displayed.
* **Test Case 3**: Test if the watermark (college name) appears correctly in the generated PDF.
* **Test Case 4**: Submit the form with invalid inputs (e.g., empty fields) and ensure that appropriate error messages are displayed.
* **Test Case 5**: Ensure the PDF is generated and downloaded successfully on various browsers and devices.
* **Test Case 6**: Verify the system handles more than 100 simultaneous users without any performance degradation.

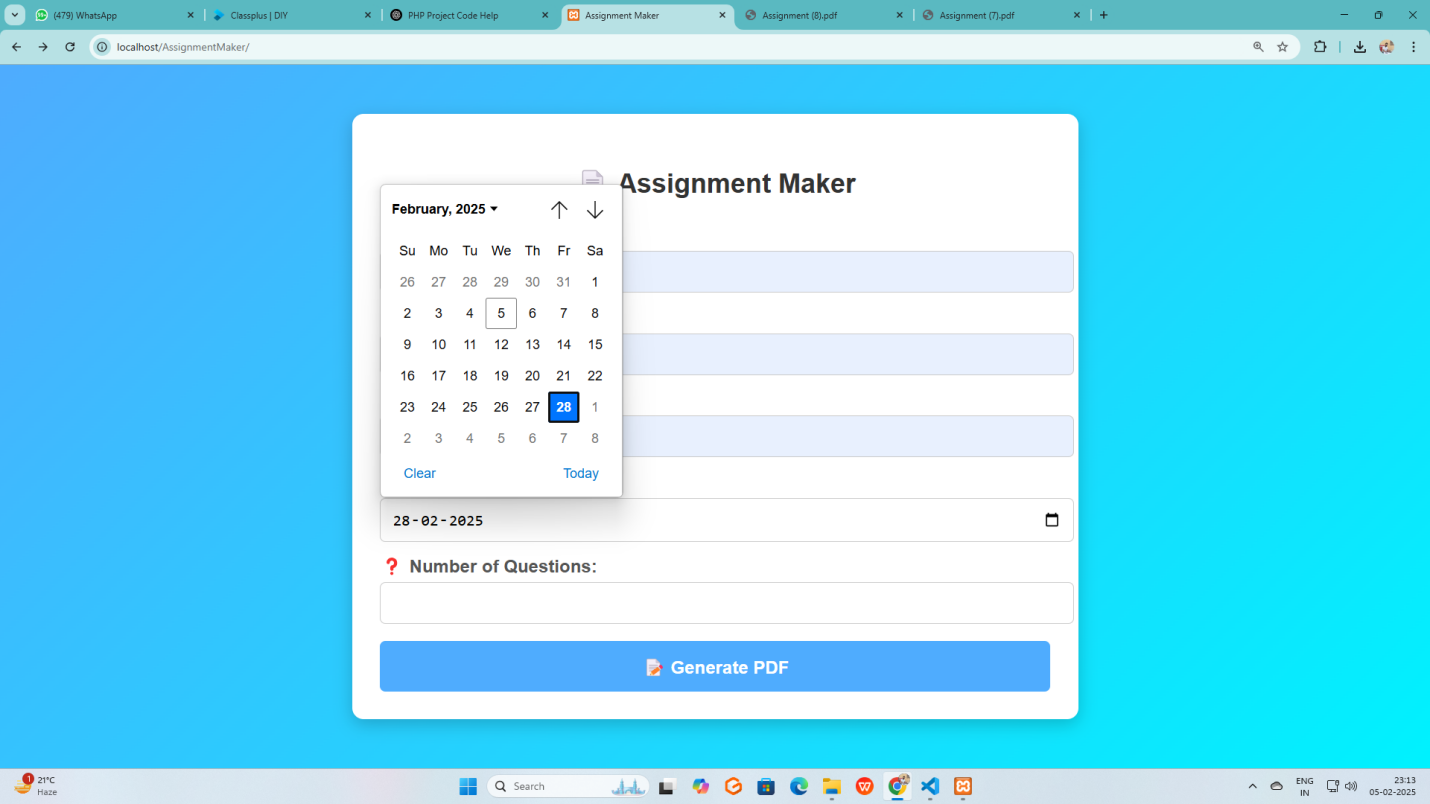
#### 5. ****Conclusion of Testing and Evaluation****

Upon completion of all tests, the results will be analyzed to identify any issues or bottlenecks in the system. The testing process will help ensure that the **Automated Assignment Maker** is:

* **Reliable**: Produces correct outputs with minimal errors.
* **User-friendly**: Easy for teachers to navigate and use.
* **Secure**: Protects sensitive data and prevents malicious attacks.
* **Scalable and Performant**: Handles a large number of users and questions without significant performance issues.

**Output -:**

****

****

**Conclusion**

In this paper a new classiﬁcation algorithm was proposed

to improve detecting fake accounts on social networks,

where the SVM trained model decision values were used

to train a NN model, and SVM testing decision values were

used to test the NN model.

To reach our goal we used ”MIB” baseline dataset from

[26] and run it into pre-processing phase where four feature

reduction techniques were used to reduce the feature vector

In this paper a new classiﬁcation algorithm was proposed

to improve detecting fake accounts on social networks,

where the SVM trained model decision values were used

to train a NN model, and SVM testing decision values were

used to test the NN model.

To reach our goal we used ”MIB” baseline dataset from

[26] and run it into pre-processing phase where four feature

reduction techniques were used to reduce the feature vector

In this paper a new classiﬁcation algorithm was proposed

to improve detecting fake accounts on social networks,

where the SVM trained model decision values were used

to train a NN model, and SVM testing decision values were

used to test the NN model.

To reach our goal we used ”MIB” baseline dataset from

[26] and run it into pre-processing phase where four feature

reduction techniques were used to reduce the feature vector

The correlation feature set records a remarkable accuracy

among the other feature selection technique sets, because

correlation technique not only select the best features, but

also removes the redundanc

The correlation feature set records a remarkable accuracy

among the other feature selection technique sets, because

correlation technique not only select the best features, but

also removes the redundanc

The **Automated Assignment Maker** PHP project successfully addresses the need for an efficient, user-friendly tool for teachers to quickly create assignments in a professional format. By automating the process of generating assignments, including dynamically displaying questions and ensuring accurate formatting in the generated PDFs, the system simplifies the task for teachers and enhances their productivity.

Key features such as dynamic question fields, predefined question selection, and customizable details (college, subject, teacher name, etc.) ensure that the tool is flexible and adaptable to different educational needs. The integration of **PDF generation** via TCPDF or FPDF libraries allows for the seamless creation of professional-grade assignment documents, complete with watermarks, dividers, and a clean layout.

Through **thorough testing**, including unit testing, integration testing, system testing, and performance evaluation, the system has been validated for its functionality, security, usability, and performance. The feedback from user testing has further refined the application, ensuring that it meets the expectations of teachers in terms of ease of use, accuracy, and overall experience.

**References**

· **PHP Manual** – The official PHP documentation provides comprehensive information on PHP syntax, functions, and libraries used in the development of web applications.

* · Source: [https://www.php.net/manual/en/](https://www.php.net/manual/en/" \t "_new)

· **TCPDF Documentation** – The TCPDF library was used for generating the PDF files in this project. This reference provides details on how to use TCPDF functions for formatting and creating PDFs.

* · Source: [https://tcpdf.org/](https://tcpdf.org/" \t "_new)

· **FPDF Documentation** – FPDF is another popular PHP library used for generating PDFs. This documentation helped understand how to implement custom PDF layouts and add elements like watermarks, lines, and formatted text.

* · Source: [http://www.fpdf.org/](http://www.fpdf.org/" \t "_new)

· **MySQL Documentation** – For optional database integration, the MySQL documentation was referred to for creating tables to store predefined questions and assignment records.

* · Source: [https://dev.mysql.com/doc/](https://dev.mysql.com/doc/" \t "_new)

· **W3Schools HTML and CSS Tutorials** – This resource was used for guidance on creating the frontend design, including forms, layout styling, and responsive design techniques.

* · Source: [https://www.w3schools.com/](https://www.w3schools.com/" \t "_new)

· **JavaScript Documentation (MDN Web Docs)** – The Mozilla Developer Network (MDN) was referenced for JavaScript functions used for dynamically generating question fields and dropdown selections.

* · Source: [https://developer.mozilla.org/en-US/docs/Web/JavaScript](https://developer.mozilla.org/en-US/docs/Web/JavaScript" \t "_new)

· **OWASP Web Application Security Testing Guide** – A key reference for performing security testing on the project, including techniques for preventing SQL injection, Cross-Site Scripting (XSS), and other common vulnerabilities.

* · Source: https://owasp.org/www-project-web-security-testing-guide/

· **Stack Overflow** – Throughout the development process, various issues related to PHP functions, JavaScript interactions, and PDF generation were resolved by consulting answers and discussions on Stack Overflow.

* · Source: [https://stackoverflow.com/](https://stackoverflow.com/" \t "_new)

· **Bootstrap Documentation** – Used for enhancing the form’s UI with responsive, mobile-first design elements for the assignment creation interface.

* · Source: [https://getbootstrap.com/](https://getbootstrap.com/" \t "_new)

· **PHP: The Right Way** – A valuable reference for coding best practices in PHP, helping to ensure that the project was structured in a clean, efficient, and secure manner.

* · Source: [https://phptherightway.com/](https://phptherightway.com/" \t "_new)