**Software Requirement Specification for**

**Valuation Automation**

|  |  |
| --- | --- |
| **Name** | SHEHANAZ D |
| **Roll no** | 7376222CT149 |
| **Seat no** | 174 |
| **Project ID** | 32 |
| **Problem Statement** | Valuation Automation |

**1. Problem Statement:**

Develop an automated faculty valuation system to streamline payment calculations for roles such as Chief Examiner, Board Chairman, Tabulator and Data Entry Operator. The system will accurately compute payments based on script counts, sessions, and script types (UG/PG). Inputs include Faculty ID, foil card allocation, and course details, with outputs showing course code, paper costs, number of papers, and total cost.

**2. Project Flow:**

**Gathering of Input:**

* Gather the course code, faculty ID, foil card distribution, and script counts (UG/PG).
* Collect all the information required to complete payment calculations.

**Data processing:**

* Get role-specific rates and faculty information from the database.
* Determine the payment for every position by utilizing session data and script counts.

**Output Generation:**

* Determine the total payment, the quantity of papers, and the cost of paper.
* Compile and arrange data to ensure a comprehensible display.

**Results Shown:**

* Give the user access to the course code, paper costs, quantity of papers, and total amount paid.
* Make sure all of the information is clear and accurate.

**Finish:**

* Complete and shut down the procedure.
* If necessary, provide alternatives for additional actions.

**3. Functional Requirements:**

**Input Handling:**

* System must accept Faculty ID, foil card allocation, course code, and UG/PG script counts.
* Validate input data for accuracy and completeness.

**Payment Calculation:**

* Calculate payments for each role (Chief Examiner, Board Chairman, Data Entry Operator, Examiner, Tabulator) based on predefined rules.
* Handle special cases for minimum and maximum script counts as specified.

**Data Retrieval:**

* Fetch faculty details and role-specific payment rates from the database.
* Retrieve and process foil card allocation and script counts.

**Output Generation:**

* Generate and display course code, cost of paper, number of papers, and total payment cost.
* Ensure outputs are correctly formatted and easy to interpret.

**User Interface:**

* Provide a user-friendly interface for input entry and result display.
* Allow users to review and verify calculations before finalizing.

**4. Non-Functional Requirements:**

1. **Performance:**
   * The system must process inputs and generate results within 2 seconds to ensure a responsive user experience.
   * Handle up to 500 transactions per minute without degradation in performance.
2. **Reliability:**
   * Ensure the system has an uptime of 99.9% to guarantee availability.
   * Implement robust error handling and recovery mechanisms to minimize downtime.
3. **Scalability:**
   * Design the system to accommodate increased load and additional roles or payment rules in the future.
   * Ensure the database and application can scale horizontally or vertically as needed.
4. **Security:**
   * Protect sensitive data, including faculty details and payment information, with encryption and access controls.
   * Implement authentication and authorization to restrict access to authorized users only.
5. **Usability:**
   * Provide an intuitive and user-friendly interface for data entry and result review.
   * Include clear instructions and help options to assist users in navigating the system.

**5.Technical Requirement:**

|  |  |
| --- | --- |
| Front End | • HTML  • CSS  • JS |
| Back End | • Python  • Django(Python Web) |
| Database | • PostgreSQL  • MySQL |
| API | • OpenAPI  • SOAP APIs  • REST Ful API |

**FLOW CHART:**

