**Title of the Project**
Agent Based Push Mechanism.

**Abstract of the project**
This project presents the development and implementation of an Agent-Based Push Mechanism aimed at improving the efficiency and personalization of information delivery in distributed systems. The core of the system relies on intelligent software agents that monitor user behavior, preferences, and environmental context to autonomously trigger relevant push notifications or actions. Unlike traditional pull-based systems where users must request information, this push mechanism enables proactive engagement by allowing agents to make real-time decisions and collaborate with other agents in the network. The system is designed with modularity, scalability, and adaptability in mind, making it suitable for applications such as smart notifications, recommendation systems, and context-aware services. The project includes a prototype implementation, performance analysis, and evaluation of the mechanism's effectiveness in enhancing user experience and reducing unnecessary data exchanges.

**Keywords**
**Generic Keywords**
Databases, Programming
**Specific Technology Keywords**
HTML, CSS, JavaScript, PHP
**Project Type keywords**
Agent-Based Systems, Push Notification Mechanism, Real-Time Communication, Recommendation Systems

**Functional components of the project**
The Agent-Based Push Mechanism project includes components for user behavior monitoring, autonomous agents for decision-making, personalized push notifications, adaptive learning, real-time communication, and user interface. It ensures scalability, security, and privacy, optimizing proactive, context-aware notifications to enhance user experience and engagement.

**Users of the system:** Authorized User and Administrator are the users of this system.

**Functionality:**
A breakdown of the functionality of Agent-Based Push Mechanism project with login/sign-up, feedback, genre selection, and an admin page:

## 1. Login and Sign-Up Page

- **User Authentication**:
  **Login**: Allows users to securely log in using their credentials (email, username, or social media accounts).
  **Sign-Up**: New users can create an account by providing necessary details (email, username, password) and agreeing to privacy terms.

## 2. Genre Selection

- **User Preferences**:
  Once logged in, users can choose from a variety of genres (e.g., music, movies, books, news). The system dynamically adapts to the user's selection and customizes push notifications and content based on their chosen genres.

- **Genre Personalization**:
  Users can update their genre preferences at any time, allowing agents to adjust content delivery accordingly.
  The agent learns user preferences over time and fine-tunes notifications based on interactions within selected genres.

## 3. Feedback Page

- **User Feedback Collection**:
  After receiving notifications, users can provide feedback on each notification (e.g., "Helpful", "Not Interested", "Dismiss").
  Users can rate notifications or content (e.g., thumbs up/down, star ratings).

- **Feedback Submission**:
  Users can submit detailed feedback on content (e.g., suggestions for improvement, complaints, or feature requests).

- **User Satisfaction Monitoring**:
  The system tracks user feedback and adjusts the agent's decision-making to improve future notifications.

## 4. Admin Page

- **User Activity Monitoring**:
  Admins can view detailed logs of user activities, including login history, notification interactions, genre preferences, and feedback submissions.
  Admins can track the effectiveness of push notifications by monitoring user engagement metrics (e.g., open rates, click-through rates).

- **Feedback Review**:
  Admins can review and categorize user feedback to identify common issues or areas for improvement.
  Admins can reply to user feedback or provide updates on issues that have been addressed.

- **User Management**:
  Admins can manage user accounts (e.g. reset passwords, or resolve issues).

- **Content and Notification Control**:
  Admins can manage content, update push notification templates, or adjust notification strategies based on user preferences and feedback.

- **Analytics Dashboard**:
  Provides insights into user engagement, genre popularity, notification performance, and user feedback trends.
  Helps optimize notification delivery and overall user experience.

## 5. Push Notification Engine

- **Personalized Notifications**:
  Based on user genre preferences and interactions, agents send tailored push notifications (e.g., new content, offers, reminders).

- **Adaptive Learning**:
  The push mechanism uses feedback data to improve notifications, ensuring content is more relevant to   the user.
- **Real-Time Delivery**:
  Notifications are triggered based on real-time user activity, context (e.g., location, device), and preferences.
- **Frequency Control**:
  Users can manage how often they receive notifications, and the system adapts to these settings.

## 6. Summary of User Flow:

- **User Login/Sign-Up**: Users authenticate themselves via the login/sign-up page.
- **Genre Selection**: Upon successful login, users select preferred genres for personalized notifications.
- **Feedback**: After receiving notifications, users provide feedback to fine-tune the content delivery.
- **Admin Monitoring**: Admins use the admin page to monitor user activity, manage content, and respond to user feedback.

### Steps to start-off the project:

Microsoft platform: The system is developed using Active Server Pages as the  front end and PHP, SQL Server/DB2 as the back end.

### The following steps will be helpful to start off the project :-

1. Get a firm grasp on the above technology.
2. Get the domain knowledge.
3. Using PHP, MySQL
4. Decide on the number of users and their profile
5. Help should be very user friendly.
6. UI should include good content and have a constant look and feel throughout the application.

### Requirements :-

### Hardware requirements –

| Number | Description | Alternatives (If available) |
|--------|-------------|------------------------------|
| 1 | PC with 512GB hard-disk and 8GB RAM | Not-Applicable |

**Software requirements -**

| Number | Description | Alternatives (If available) |
|--------|-------------|----------------------------|
| 1 | Windows11 with MS-office | Not Applicable |
| 2 | PHP | |
| 3 | MS-SQL server | Oracle |
| 4 | Windows | Linux |

**Manpower requirements**

2 students can complete this in 4 – 6 months if they work fulltime on it.

**Milestones and Timelines**

| Number | Milestone Name | Milestone Description | Timeline<br><br>Week no. from the Start of the project | Remarks |
|--------|----------------|----------------------|-------------------------------------------------------|---------|
| 1 | Requirements Specification | User authentication, genre selection, push notifications, feedback system, admin dashboard, real-time processing, security, scalability, responsive design. | 2-3 | Attempt should be made to add some more relevant functionalities other than those that are listed in this document. |
| 2 | Technology familiarization | Understanding of the technology needed to implement the project. | 4-5 | The presentation should be from the point of view of being able to apply it to the project, rather than from a theoretical perspective. |
| 3 | Database creation | Design schema, create tables, store user data, feedback, preferences. | 5-7 | It is important to finalize on the database at this stage itself so that development and testing can proceed with the actual database itself. |

| 4 | Implementation of the front-end of the system | Set up project, implement authentication, genre selection, feedback system, admin dashboard, Real time notifications, UI enhancements, testing, deployment. | 10-12 | During this milestone period, it would be a good idea for the team (or one person from the team) to start working on a test-plan for the entire system. This test-plan can be updated as and when new scenarios come to mind. |
|---|---|---|---|---|
| 5 | Integrating the front-end with the database | The front end developed in the earlier milestone will now be able to update the database. Other features like. In short, the system should be ready for integration testing. | 12-13 | During this period Integrating the frontend with the backend it is a fundamental step in developing a web application |
| 6 | Integration Testing | The system should be thoroughly tested by running all the testcases written for the system (from milestone 5). | 14-15 | Another 2 weeks should be there to handle any issues found during testing of the system. After that, the final demo can be arranged. |
| 7 | Final Review | Issues found during the previous milestone are fixed and the system is ready for the final review. | 16-18 | During the final review of the project, it should be checked that all the requirements specified during milestone number 1 are fulfilled (or appropriate reasons given for not fulfilling the same) |

**Guidelines and References**
https://www.w3schools.com/php/  (PHP tutorial)
https://javascript.info/   (Javascript tutorial)
https://www.w3schools.com/sql/sql_server.asp  (SQL-server tutorial)