**ANALYSIS AND DESIGN**

**INFORMATION SYSTEM FOR**

**MyUni**

[**System Request 3**](#_qqkjdcrwv7m)

[**Feasibility Study 4**](#_fzzr8939zm5i)

[**ANALYSIS 5**](#_py0ffumosy7t)

[BUSINESS PROCEDURES 5](#_m8cyjuzdsi95)

[UML activity diagrams of the state of the new information system (AS-IS) 6](#_plccgd7dmtyo)

[UML activity diagrams of the state of the new information system (TO-BE) 7](#_3tu25avahx03)

[User Stories 9](#_fkhn6fcdmz3q)

[EPICS 10](#_btraw6lmrsy8)

[**Initial Information System Backlog 11**](#_f7ou9bb3nafl)

[**Requirements analysis 13**](#_kqxjl0ex36zi)

[Functional 13](#_72ena4ux3ocb)

[Non functional 14](#_qy9vkup1027s)

[Operational 15](#_53fh2kd60grp)

[Performance 16](#_cyj68yhrtg73)

[Security 16](#_6ar4gxhy38ul)

[Cultural & Political 17](#_twy674ld4zuw)

[UML use case diagram 18](#_tn1gpnnoeyjn)

[Verbal descriptions and indicative screens 18](#_1zknd3cv09x)

## **System Request**

|  |  |
| --- | --- |
| Name of the project | **MyUni** |
| Project sponsor | **GFOSS (Greek Free/Open Source Software Society)** |
| Business Need | An integrated platform to streamline administrative and academic processes in universities. |
| Business Requirements | • Secure user authentication and profile management • Real-time academic data and scheduling • Flexible CMS for fast updates • Multi-domain support and seamless integrations • Responsive design for all devices |
| Business Value | •Greater efficiency and time savings•Enhanced user experience and engagement•Informed, data-driven decision-making• Scalability and competitive advantage |
| Specific issues or limitations | • Challenges with legacy data migration• User adoption and training needs • Regulatory and compliance requirements |

##

## **Feasibility Study**

|  |  |
| --- | --- |
| Technical feasibility | **Frontend:** React.js with TypeScript, Vite, TailwindCss, Motion framer, React icons(for UI/UX and dynamic interaction and Robust performance).**Backend:** Node.js with Express.js (handling API requests).**Database:** PostgreSQL or MongoDB (for student, course, and admin data, Realtime connection).**Development Approach:** Adopt Agile software development for iterative delivery and continuous improvement. |
| Financial feasibility |  **Potential Costs:** * Server hosting & cloud services.
* Development team salaries (if outsourced or in-house).
* Third-party API subscriptions (email notifications, online payments).

 **Funding Sources:** * University budget.
* Government or educational grants.
* Potential partnerships with tech companies.
 |
| Corporate feasibility | **• Stakeholder Acceptance:** Universities, faculty, and students need to see clear benefits and receive proper training **• Integration Feasibility:** Must integrate smoothly with any existing systems  |
| Legal feasibility | • Compliance with data protection laws (e.g., GDPR), accessibility standards (WCAG), and intellectual property guidelines |

## **ANALYSIS**

### **BUSINESS PROCEDURES**

1. Home Page .
2. User Registration(backend integration).
3. User Login.
4. Profile Management.
5. Data Viewing and Personalisation.
6. CMS Data Management.
7. Admin Panel Management.
8. Student Portal information.
9. Sharing application instances for multiple domains.
10. Real time course Time table/Schedule of Students.
11. Integrating Students Portal database (<https://sso.uom.gr/login>).
12. Adding a section of exam results and marks for students.
13. Student Academics Material/notes.
14. Expanding Exam Schedule Design.
15. News & Announcements portal with notifications.
16. Option to Select All Three Degree Types (Undergraduate, Masters, and PhD, if applicable).
17. Lectures Recordings Portal.
18. Additional Theme Options.
19. Culture & Politics Module with detailed Policies.

### **UML activity diagrams of the state of the new information system (AS-IS)**



### **UML activity diagrams of the state of the new information system (TO-BE)**

Below are the UML activity diagrams of the state of the new information system.

* UML Activity diagrams for the following processes:
* User Registration and Login
* Profile Management
* Data Viewing and Personalization
* CMS Data Management
* Admin Panel Management
* Student Portal Information
* Sharing application instances for multiple domains

**User Registration and Login**

For full view – Ctrl + right Click



### **Student Portal UML**

### **User Stories**

**USER**

1. As a user I want to register to have access to all the features of the application.
2. As a user I want to login to my account to view personalized information.
3. As a user I want to be able to change my password
4. As a user I want to manage my profile to ensure I can see appropriate personalized information.
5. As a user I want to use the application on my Mobile, Tablet or Desktop.

**ADMIN**

1. As an Admin I want to login to manage all the students information, and website design.
2. As an Admin I want to be able to manage a CMS for fast-changing information.
3. As an Admin I want to create website pages or design through the backend.
4. As an Admin I want to change the color schemes for the different instances.
5. As an Admin I want to control which type of data is fetched from each source.
6. As an Admin I want to be able to have multiple users with different access levels.
7. As an Admin I want to be able to reset the password of the users.

**EPICS**

**EPIC: User Authentication and Management**

1. User Story: As a user, I want to register for an account with my university credentials so that I can access personalized services.
2. User Story: As a user, I want to log in securely to my account so that my personal data is protected.
3. User Story: As a user, I want to manage my profile so that I can update my personal information.

**EPIC: CMS Data Management**

1. User Story: As an admin, I want to manage a CMS for fast-changing information.
2. User Story: As an admin, I want to create website pages or design through the backend.
3. User Story: As an admin, I want to define and schedule the publication of content.

**EPIC: Student Portal information**

1. User Story: As a student, I want to view my grades so that I can understand my performance in my courses.
2. User Story: As a student, I want to register for courses to be prepared for the upcoming semesters.
3. User Story: As a student, I want to be informed for campus events.

**EPIC: Theme and Style Management**

1. User Story: As an admin, I want to change the color schemes for the different instances.
2. User Story: As an admin, I want to manage all the assets of the page.

**EPIC: Data Configuration Management**

1. User Story: As an Admin I want to control which type of data is fetched from each source.
2. EPIC: Users Access Levels Management
3. User Story: As an Admin I want to be able to have multiple users with different access levels.

**EPIC: Access management**

User Story: As an Admin I want to be able to reset the password of the users.

### **Initial Information System Backlog**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| EPIC | STORY\_ID | Description Story | MoSCoW Priority  | Dependencies | Ideal days estimate | Sprint |
| User Authentication and Management | 1 | As a user, I want to register for an account with my university credentials. | Must have | None | 3 | 1 |
| User Authentication and Management | 2 | As a user, I want to log in securely to my account. | Must have | Story 1 | 2 | 1 |
| User Authentication and Management | 3 | As a user, I want to manage my profile. | Should have | Story 1, Story 2 | 3 | 2 |
| CMS Data Management | 4 | As an admin, I want to manage a CMS for fast-changing information. | Must have | None | 5 | 3 |
| CMS Data Management | 5 | As an admin, I want to create website pages or design through the backend. | Should have | Story 4 | 8 | 5 |
| CMS Data Management | 6 | As an admin, I want to define and schedule the publication of content. | Must have | Story 4 | 5 | 3 |
| Student Portal information | 7 | As a student, I want to view my grades. | Must have | Story 2 | 3 | 2 |
| Student Portal information | 8 | As a student, I want to register for courses. | Should have | Story 2 | 5 | 3 |
| Student Portal information | 9 | As a student, I want to be informed for campus events. | Must have | None | 2 | 1 |
| Theme and Style Management | 10 | As an admin, I want to change the color schemes for the different instances. | Must have | Story 4 | 3 | 2 |
| Theme and Style Management | 11 | As an admin, I want to manage all the assets of the page. | Should have | Story 4 | 5 | 3 |
| Data Configuration Management | 12 | As an Admin I want to control which type of data is fetched from each source. | Must have | None | 3 | 2 |
| Users Access Levels Management | 13 | As an Admin I want to be able to have multiple users with different access levels. | Must have | None | 3 | 2 |
| Access management | 14 | As an Admin I want to be able to reset the password of the users. | Must have | None | 2 | 1 |

### **Requirements analysis**

#### **Functional**

|  |  |  |
| --- | --- | --- |
| **Α/Α** | **Requirement** | **High/Medium/Low** |
| **1 User Authentication** |
| 1.1 | User registration using university credentials. | High |
| 1.2 | Secure login process with password encryption. | High |
| 1.3 | Password reset functionality. | High |
| **2. Profile Management** |
| 2.1 | Update personal information (name, email, etc.). | High |
| 2.2 | Manage notification preferences. | Medium |
| **3. CMS Data Management** |
| 3.1 | Create new content pages. | High |
| 3.2 | Edit existing content pages. | High |
| 3.3 | Schedule content publication. | High |
| **4. Student Portal Information** |
| 4.1 | Display student grades. | High |
| 4.2 | Course registration functionality. | High |
| 4.3 | Display campus events. | High |
| **5. Theme and Style Management** |
| 5.1 | Customize application color scheme. | High |
| 5.2 | Upload university logo and branding assets. | High |
| **6. Users Access Levels Management** |
| 6.1 | Access configuration levels | High |
| 6.2 | Reset user Password | High |
| **7. Data Configuration Management** |
| 7.1 | Select source of data to be fetched from each source. | High |

#### **Non functional**

##### **Operational**

|  |  |  |
| --- | --- | --- |
| **Α/Α** | **Requirement** | **High/Medium/Low** |
| 1 | Usability: The system should be easy to use and navigate, with a user-friendly interface. | High |
| 2 | Reliability: The system must operate without errors or interruptions. | High |
| 3 | Adaptability: The user interface should be adaptive for use on mobile devices and tablets. | High |
| 4 | Scalability: The system must be scalable to support additional functions in the future. | Medium |
| 5 | Access management: the system must have capabilities to manage user roles and rights. | High |
| 6 | Interoperability: The system must be compatible and able to integrate with other systems and APIs. | Medium |
| 7 | Environmental performance: the system must be optimised for minimum resource consumption. | Low |
| 8 | Monitoring and logging: The system shall provide monitoring and logging capabilities to diagnose problems. | Medium |

##### **Performance**

|  |  |  |
| --- | --- | --- |
| **Α/Α** | **Requirement** | **High/Medium/Low** |
| 1 | Response time: The application must respond in <2 seconds for all basic operations (e.g., user login, search for actions). | High |
| 2 | Loading Page: Most pages of the application should load in <1 second. | High |
| 3 | Concurrent Users: The application must support up to 5000 concurrent users without performance degradation. | High |
| 4 | Recovery Time from Error: The application must recover from errors within 5 seconds. | High |
| 5 | Data processing: processing large volumes of data (e.g. usage statistics) should be completed in <5 seconds. | Medium |
| 6 | Search efficiency: search results should be displayed in <1 second. | High |
| 7 | Interface Reaction: The user interface must react directly to user actions (e.g., clicking buttons, changing pages). | High |
| 8 | Cache Memory Performance: The use of cache memory should be optimized to minimize requests to the server. | Medium |
| 9 | Data Update Speed: Data updates (e.g. form submission, feedback) should be completed in <2 seconds. | Medium |
| 10 | Resource Efficiency: the application must be resource efficient (CPU, memory) to support optimal performance. | Medium |
| 11 | Disaster Recovery Time: The application must recover its data and operations within 4 hours of catastrophic events. | High |
| 12 | Minimize User Difficulty: Users should not experience delays or glitches when navigating the app. | High |
| 13 | Support for Multilingualism: The performance of the application should not be affected by supporting multiple languages. | Low |
| 14 | Upgradeability: The application must be able to be upgraded without interrupting performance. | Medium |
| 15 | Performance monitoring: a performance monitoring system should be in place to diagnose problems and improve performance. | Medium |
| 16 | Availability: The system must be available 99.9% of the time. | High |
| 17 | Maintenance: The system shall be designed for easy maintenance and upgrades. | Medium |

##### **Security**

|  |  |  |
| --- | --- | --- |
| **Α/Α** | **Requirement** | **High/Medium/Low** |
| 1 | User authentication: The application must use secure authentication methods (e.g., OAuth, 2FA). | High |
| 2 | Data encryption: all sensitive data must be encrypted during storage and transfer. | High |
| 3 | Data Security: user data must be stored and transmitted using encryption (e.g. SSL/TLS). | High |
| 4 | SQL Injection Prevention: the application must be protected against SQL injection attacks through appropriate input processing. | High |
| 5 | Secure Password Management: User passwords must be stored using secure hashing algorithms (e.g. bcrypt). | High |
| 6 | Session Management: User sessions should automatically expire after a predefined period of inactivity. | High |
| 7 | Strong Code Policy: Users must create codes that meet specific complexity requirements. | High |
| 8 | Secure Password Recovery: The password recovery process must include secure authentication steps. | High |
| 9 | Secure API Access: all API calls must be secure and require authentication. | High |
| 10 | Recording and Monitoring: The application must record all significant user actions and support the tracking of these. | Medium |
| 11 | Access Control: Users should only have access to the functions and data they are authorized to view or edit. | High |
| 12 | Mobile Data Security: Data transferred on mobile devices must be encrypted and protected. | Medium |
| 13 | Preventing Malicious Activities: the application must detect and prevent potentially malicious activities (e.g. brute force attacks). | High |
| 14 | Network Security: The application must be protected from network attacks by firewall and other network security methods. | High |
| 15 | Ensuring Data Integrity: the application data must be secured so that it is not corrupted or destroyed. | Medium |
| 16 | Penetration tests: The application shall be subjected to regular penetration tests to identify and repair vulnerabilities. | Medium |
| 17 | Privacy Policy: The application must have a clear privacy policy that informs users about the management of their data. | High |
| 18 | Secure Software Updates: Software updates must be performed in a secure manner to avoid introducing new vulnerabilities. | Medium |

#####

##### **Cultural & Political**

|  |  |  |
| --- | --- | --- |
| **Α/Α** | **Requirement** | **High/Medium/Low** |
| 1 | Multilingual Support: The application must support multiple languages, starting with Greek and English, to accommodate different users. | High |
| 2 | Accessibility: The application must be accessible to people with disabilities, following WCAG 2.1 AA standards. | High |
| 3 | GDPR compliance: the application must comply with the EU General Data Protection Regulation (GDPR). | High |
| 4 | Diversity and Inclusion: The application must support diversity and inclusion, ensuring that content and interaction is appropriate for all users. | Medium |
| 5 | Avoiding Hate Content: The application must prevent the publication and dissemination of content that promotes hatred, violence or discrimination. | High |
| 6 | Compliance with the ePrivacy Directive: the application must comply with the ePrivacy Directive. | Medium |
| 7 | Transparency: The application must provide clear and transparent information on the use of users' data. | High |
| 8 | User Participation Policy: The application should encourage active user participation in polls and surveys to improve services. | Low |
| 9 | Ethical Use of Data: the application must ensure that users' data is used ethically and only for the purposes stated. | High |
| 10 | Support for Local Laws: the application must comply with local laws and regulations in each region where it operates. | High |
| 11 | User Content Policy: The application should have a policy for managing user-generated content, ensuring that it is appropriate and secure. | Medium |
| 12 | User Education: The application should provide resources and guidance to educate users on security and usage policies. | Low |
| 13 | Crisis management: the application must have a crisis management plan to deal with possible emergencies or security breaches. | Medium |
| 14 | User Acceptance Policy: The application should include clear terms and conditions for its use, as well as procedures for dealing with violations. | High |
| 15 | Anonymity policy: The application must allow users to use the services anonymously where possible and appropriate. | Medium |
| 16 | Policy flexibility: The application must be able to adapt to changes in policies and regulations, incorporating new requirements quickly and efficiently. | Medium |
| 17 | Consent Management Policy: The application must manage user consent for data collection and use by providing easy procedures for withdrawing consent. | High |
| 18 | Cooperation with Supervisory Authorities: The application shall cooperate with supervisory authorities to ensure compliance with regulations. | Medium |
| 19 | Informing Users about Policy Changes: Users should be notified in a timely manner of any changes to the application's policies. | Medium |

#### **UML use case diagram**

Below is the UML use case diagram. --- do I need to make these also?

#### **Verbal descriptions and indicative screens**

**1. Home Page**

**Description:**
The home page serves as the landing screen for visitors and users, providing an overview of the platform along with key navigation elements and dynamic content.

**Regular Flow:**

1. The visitor or user enters the platform URL.
2. The system loads the home page featuring dynamic content such as featured announcements, news, and navigation menus.
3. The home page displays primary options, including "Register" and "Login" for new or returning users.
4. If a user is already authenticated, the system greets them by name and shows personalized shortcuts and notifications.

**Alternative Flow:**

* If the system experiences slow response times, a loading indicator is shown until the content is fully loaded.
* If an error occurs while loading the home page, the system displays an error message with an option to retry.

**Sub-Flow:**

* If a visitor has an active session, the system may automatically redirect them to their personalized dashboard instead of the standard home page.

**Indicative Screen:**

* A welcoming banner at the top with a brief platform introduction.
* Prominent “Register” and “Login” buttons near the center for quick access.
* Dynamic panels (tiles) displaying announcements or upcoming events.
* If logged in, a personalized greeting and shortcuts (e.g., “Go to Your Profile,” “View Courses”) appear at the top.

**2. User Registration**

**Description:**

The user registers on the platform by creating a new account.

**Regular Flow:**

1. The user selects "Register".
2. Fill in the required fields: email, first name, last name, city of residence, password.
3. Confirms that he/she agrees with the terms and conditions.
4. The user clicks on the "Register" button.
5. The system checks if all fields are filled in correctly and if the email has not been used before.
6. The system creates the new account and sends a confirmation email to the user.

**Alternative Flow:**

* If the data is not filled in correctly or the email has been used, the system displays the message "The data is not correct. Please correct it and try again".
* The UC continues from step 2 of the regular flow.

**Sub-Flow:**

* If the user selects "Cancel", the user is returned to the home page.

**Indicative Screen:**

* A registration form with labeled text fields for email, name, etc.
* A checkbox for “I agree to the terms and conditions” followed by a link to view them.
* A clear “Register” button, and a “Cancel” link that returns the user to the home page.
* Real-time validation feedback (e.g., red highlight if email is already taken).

**3. User Login**

**Description:**

User logs in.

**Regular Flow:**

1. User clicks "Login."
2. User enters credentials (email, password).
3. User clicks "Submit."
4. System verifies credentials and logs the user in.

**Alternative Flow:**

* If the data is wrong, the system displays an error message and asks the user to correct it.
* The UC continues from step 2 of the regular flow.

**Sub-Flow:**

* If the user selects "Reset password", the uses goes to the reset password procedure (UC 3).

**Indicative Screen:**

* A simple login page with two fields (email, password) and a “Submit” button.
* A “Forgot Password?” or “Reset Password” link below the password field.
* An optional “Remember Me” checkbox for convenience.

**4. Reset Password**

**Description:**

The user resets his password when he does not remember the login credentials.

**Regular Flow:**

1. The user receives an error message when logging into their account (UC 2).
2. The user selects "Reset password".
3. The user fills in their email.
4. The user clicks the "Send" button.
5. The system checks the email and, if it is correct, sends a password reset email to the user.

**Alternative Flow:**

* If the email is not correct, the system displays the message "Your email is not correct. Please enter the correct email".
* The UC continues from step 3 of the regular flow.

**Sub-Flow:**

* If the user selects "Cancel", the user is returned to the login page.

**Indicative Screen:**

* A simple login page with two fields (email, password) and a “Submit” button.
* A “Forgot Password?” or “Reset Password” link below the password field.
* An optional “Remember Me” checkbox for convenience.

**5. User Profile Management**

**Description:**

The user manages his profile by updating his personal information.

**Regular Flow:**

1. The user logs in to his/her account (UC 2).
2. The user selects "Profile".
3. The user updates the fields: first name, surname, city of residence, profile photo.
4. The user presses the "Save" button.
5. The system updates the profile data and displays a confirmation message.

**Indicative Screen:**

* A profile page showing user details (name, city, profile photo) with editable text fields.
* A “Save” button at the bottom, which on click triggers a confirmation popup or toast message.
* A small “Change Photo” button to upload or remove a profile picture.

**6. Account deletion**

**Description:**

The user deletes his/her account from the application.

**Regular Flow:**

1. The user logs in to his/her account (UC 2).
2. The user selects "Profile".
3. The system returns the user's profile.
4. The user selects "Delete Account".
5. The user confirms the action.
6. The system deletes the account and all user data.

**Alternative Flow:**

* If the user cancels the action, the user is returned to the previous page.
* The UC continues from step 3 of the regular flow.

**Indicative Screen:**

* A prominent “Delete Account” button, possibly highlighted in red.
* A confirmation dialog: “Are you sure you want to delete your account?” with “Yes” and “No” options.
* A final message if deletion succeeds, redirecting the user to the home page.

**7. CMS Content Creation**

**Description:**

Admin creates new content pages.

**Regular Flow:**

1. Admin logs in.
2. Admin navigates to CMS section.
3. Admin clicks "Create New Page."
4. Admin fills in title, content, and other metadata.
5. Admin clicks "Publish".
6. System publishes to be visible to the users.

**Indicative Screen:**

* A CMS editor interface with fields for “Title,” “Content,” and optional metadata (tags, categories).
* A “Publish” or “Save Draft” button at the top or bottom.
* A preview pane showing how the page will look before publishing.

**8. Push Notifications Management**

**Description:**

The administrator manages the platform notifications.

**Regular Flow:**

1. The administrator logs in to his/her account (UC 2).
2. The system displays the control panel.
3. The administrator selects 'Notifications'.
4. The administrator creates or edits alerts.
5. Specifies the recipients and the time of sending.
6. The system sends the notifications to the users.

**Indicative Screen:**

* A “Notifications” tab within the admin control panel.
* A form to create or edit a notification (title, message, recipients).
* A scheduling option with date/time pickers for sending notifications.
* A list view of pending or previously sent notifications.

**9. User Logout**

**Description:**

The user is logged out of the platform.

**Regular Flow:**

1. The user selects "Logout".
2. The system logs the user out and returns him to the home page.

**Indicative Screen:**

* Typically, no dedicated screen. After clicking “Logout,” the user sees a quick “You have been logged out” message (if any), then is redirected to the home page.
* Some platforms display a logout confirmation or redirect message.

**10. Reset User Password by Administrator**

**Description:**

The administrator resets a user's password.

**Regular Flow:**

1. The administrator enters his/her account (UC 2).
2. The system displays the control panel.
3. The administrator selects "Manage Users".
4. The administrator selects the user who needs a password reset.
5. The administrator presses the "Reset Password" button.
6. The system generates a new password and sends it to the user via email.

**Indicative Screen:**

* A “Manage Users” page listing all users.
* An action button next to each user entry labeled “Reset Password.”
* A confirmation popup: “Reset password for this user?”
* A success toast or message: “Password reset email sent.”

**11. User Support**

**Description:**

The administrator provides support to the users of the platform.

**Regular Flow:**

1. The administrator enters his/her account (UC 2).
2. The system returns the dashboard.
3. The administrator selects "Support".
4. A list of support requests is displayed.
5. The administrator selects a request to view details.
6. The administrator responds to the request and records the actions taken.
7. The system informs the user and closes the request.

**Indicative Screen:**

* A “Support” section with a table of open/closed tickets, each showing subject and status.
* A detail view for each request with a text box for the admin to respond.
* A “Close Request” or “Resolve” button to finalize the ticket.

**12. Integration with Social Networks**

**Description:**

The user connects their platform account with social networks for easy sharing.

**Regular Flow:**

1. The user logs in to his/her account (UC 2).
2. The user selects "Profile".
3. The user selects "Connect with Social Networks".
4. The user selects the social network and confirms the connection.
5. The system links the account and informs the user.

**Indicative Screen:**

* In the “Profile” page, a subsection labeled “Social Network Connections.”
* Icons or buttons for each supported social network (e.g., Facebook, Twitter).
* A status indicator (e.g., “Linked,” “Not Linked”).
* A confirmation message once the account is linked.

**13. Connecting to External APIs**

**Description:**

The platform connects to external APIs for additional functionality.

**Regular Flow:**

1. The administrator enters his/her account (UC 2).
2. The system returns the dashboard.
3. Admin selects "Connect with API".
4. Enter the necessary API keys and parameters.
5. The system checks the connection and informs the administrator.

**Alternative Flow:**

5.1 If the administrator has entered the wrong API key or the service is down, the system displays a message “Unable to connect to external service using API”.

5.2 The UC continues from step 4 of the regular flow.

**14. Student Portal Information**

**Description:**
The student portal is a centralized hub where students access academic records, registration details, and personalized resources.

**Regular Flow:**

1. The student logs into the platform.
2. The system loads the Student Portal dashboard, displaying sections such as “My Courses,” “Academic Records,” “Upcoming Events,” and “Notifications.”
3. The student navigates among the modules to view or update information.

**Alternative Flow:**

* If data retrieval fails, a default message is shown with a prompt to retry.

**Sub-Flow:**

* The student may personalize the dashboard by rearranging widgets or selecting preferred modules.

**Indicative Screen:**

* A dashboard view with distinct panels (tiles) for each module, a navigation bar at the top, and a sidebar for quick links to academic details.

**15. Sharing Application Instances for Multiple Domains**

**Description:**
The system supports deploying and sharing separate instances for different university domains, allowing each to customize its branding and content.

**Regular Flow:**

1. The admin logs into the Admin Panel and selects “Domain Management.”
2. The admin clicks “Create New Instance.”
3. The system prompts for a domain name, basic configuration settings, and branding details.
4. The admin confirms and the system deploys a new instance with default settings.
5. The admin customizes the instance further (e.g., layout, color scheme, content modules).

**Alternative Flow:**

* If the chosen domain name is already in use, the system displays an error message and suggests alternatives.

**Sub-Flow:**

* The admin may clone an existing instance as a template for faster deployment.

**Indicative Screen:**

* A configuration screen with input fields for domain name, a preview area for branding, and a “Clone Instance” option.

**16. Real-Time Course Timetable/Schedule of Students**

**Description:**
The system displays an up-to-date course timetable, ensuring students always have the latest schedule information.

**Regular Flow:**

1. The student logs in and selects “Course Timetable” from the dashboard.
2. The system retrieves the latest schedule data from integrated sources.
3. The timetable is presented in an interactive calendar view with filters (by course or day).
4. The student selects a course slot to view detailed information such as room number and instructor.

**Alternative Flow:**

* If real-time data is unavailable, the system displays the most recent schedule along with a “Last Updated” timestamp.

**Sub-Flow:**

* The system automatically sends push notifications if significant changes occur.

**Indicative Screen:**

* A calendar view with color-coded course entries, filter options, and a notification area showing recent updates.

**17. Adding a Section for Exam Results and Marks for Students**

**Description:**
Students can access their exam results and marks through a dedicated portal section.

**Regular Flow:**

1. The student navigates to the “Exam Results” section within the portal.
2. The system retrieves exam data (marks, comments, dates) from the academic database.
3. The results are displayed in a structured table grouped by subject or course.
4. The student can click an entry for a detailed breakdown of marks and examiner comments.

**Alternative Flow:**

* If exam data is not yet available, the system displays a message like “Results will be published soon.”

**Sub-Flow:**

* The student can choose to download or print the results for offline review.

**Indicative Screen:**

* A results dashboard with sortable columns (e.g., Subject, Score, Date), clickable rows for details, and a download/print button.

**18. Student Academics Material/Notes**

**Description:**
A dedicated section where students can access lecture notes, reading materials, and other academic resources.

**Regular Flow:**

1. The student logs in and selects “Academic Materials” from the menu.
2. The system displays a list of available materials organized by course or subject.
3. The student clicks on a material to view details or download the file.

**Alternative Flow:**

* If a particular material is not available, the system shows a “Coming Soon” notification.

**Sub-Flow:**

* The student can use filters or a search bar to quickly find specific materials.

**Indicative Screen:**

* A file repository interface with thumbnail previews, filter options (by course/type), and clear download buttons.

**19. Expanding Exam Schedule Design**

**Description:**
Administrators can design and adjust exam schedules to accommodate complex exam arrangements, conflicts, and rescheduling needs.

**Regular Flow:**

1. The admin navigates to the “Exam Schedule” module in the Admin Panel.
2. The admin inputs exam details such as dates, times, venues, and subjects.
3. The system generates a comprehensive exam schedule preview.
4. The admin reviews and confirms the schedule, which is then published for student access.

**Alternative Flow:**

* If scheduling conflicts are detected, the system alerts the admin and suggests possible resolutions.

**Sub-Flow:**

* The admin can manually override conflict resolutions and add explanatory notes for changes.

**Indicative Screen:**

* A scheduling interface with a calendar view, conflict indicators, manual adjustment options, and a confirmation button.

**20. News and Announcements Portal with Notifications**

**Description:**
A portal section where administrators can post news and announcements, with real-time notifications sent to users.

**Regular Flow:**

1. The admin logs in and selects the “News & Announcements” section from the Admin Panel.
2. The admin creates a new announcement, entering details like title, content, and scheduling information.
3. The system publishes the announcement and sends notifications (email/in-app) to all users.
4. Users can view the announcement via a notification link on their dashboard.

**Alternative Flow:**

* If required fields are missing during creation, the system prompts the admin to complete them before publishing.

**Sub-Flow:**

* Users can mark announcements as read or save them for later reference.

**Indicative Screen:**

* A content editor screen for announcements with scheduling options, a preview pane, and a notification summary on the user dashboard.

**21. Degree Selection for Multiple Programs**

**Description:**
During registration or profile updates, users select their degree type (Graduation, Masters, or PhD) to tailor their experience.

**Regular Flow:**

1. The user is prompted to select a degree program during account setup or profile update.
2. The system saves the selection and adjusts the dashboard content, notifications, and available academic resources accordingly.
3. The user proceeds with a personalized experience based on the selected degree.

**Alternative Flow:**

* If the user skips the selection, a default view is applied with a prompt to update later.

**Sub-Flow:**

* The user can change their degree selection later via the profile management module.

**Indicative Screen:**

* A dropdown or radio button selection during registration, with contextual help or examples displayed on the side.

**22. Adding Lectures Recordings Portal**

**Description:**
A dedicated portal for uploading, managing, and streaming recorded lectures for student review.

**Regular Flow:**

1. The admin logs in and navigates to the “Lecture Recordings” section in the Admin Panel.
2. The admin uploads video files along with metadata (course, title, date, description).
3. The system processes the uploads and generates preview thumbnails.
4. Students access the portal to stream or download recordings.

**Alternative Flow:**

* If an upload fails due to unsupported formats or file size, the system displays an error message with guidelines.

**Sub-Flow:**

* The system may auto-generate video descriptions based on metadata and allow the admin to edit them.

**Indicative Screen:**

* A media management interface with upload controls, a list view of recordings with thumbnails, and options for previewing and editing details.

**23. Adding More Themes Options**

**Description:**
Administrators can choose and customize from a range of visual themes to modify the look and feel of the application.

**Regular Flow:**

1. The admin accesses the “Theme Management” section via the Admin Panel.
2. A gallery of available themes is displayed, complete with preview images and basic descriptions.
3. The admin selects a theme and can customize specific elements (e.g., color schemes, fonts).
4. The admin applies the theme, and the system updates the user interface accordingly.

**Alternative Flow:**

* If the selected theme fails to load properly in preview mode, the system reverts to the current theme and displays an error message.

**Sub-Flow:**

* The admin may save custom theme settings as a new template for future use.