

Software Engineering

Software requirement specification (SRS)



March 11, 2024

beirut.tech

Beirut- Airport Road



|  |  |
| --- | --- |
| **Project name:** | Teachers' class schedule |
| **Date:** | 11/3/2024 |
| **Version:** | 1.6 |
| **By:** | Mohammad Ali Sayed Ahmad  Mohammad Obaid  Mohammad Jawad Al Jawhari |

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision history** | | | |
| **Version** | **Author** | **Version description** | **Date completed** |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Review history** | | | |
| **Approving party** | **Version approved** | **Signature** | **Date** |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Approval history** | | | |
| **Reviewer** | **Version reviewed** | **Signature** | **Date** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

[Introduction: 2](#_Toc161062474)

[feasibility study 4](#_Toc161062480)

[System Requirements: 6](#_Toc161062488)

[Functional Requirements: 6](#_Toc161062489)

[Non-Functional Requirements: 6](#_Toc161062495)

[User stories: 6](#_Toc161062502)

[As a school administrator: 6](#_Toc161062503)

[As a teacher: 6](#_Toc161062504)

[As a substitute teacher: 6](#_Toc161062505)

[Functional requirements: Use Cases Description Card: 6](#_Toc161062506)

[External Interface Requirements: 6](#_Toc161062507)

[Definitions and Acronyms: 7](#_Toc161062513)

[Choice of SDLC Model: 8](#_Toc161062523)

[Project Initiation: 8](#_Toc161062524)

[Sprint 0: 8](#_Toc161062525)

[Sprint Planning: 8](#_Toc161062526)

[Sprint Execution: 8](#_Toc161062527)

[Sprint Review: 8](#_Toc161062528)

[Sprint Retrospective: 9](#_Toc161062529)

[Repeat: 9](#_Toc161062530)

# Introduction:

## 

## Product Scope:

The product scope encompasses the development and deployment of a software solution tailored for educational institutions to streamline the scheduling and management of teacher sessions. It focuses on enhancing productivity, reducing administrative burdens, and improving overall time management within schools. With the passage of time and improvement in performance, it will be possible to meet the needs of the university and institutions interested in scheduling workers’ time programs.

## Intended Audience:

This documentation is intended for stakeholders involved in the development, implementation, and usage of the time management system. This includes software engineers, project managers, school administrators, and teachers.

## Intended Use:

The time management system is designed to assist schools in efficiently organizing and managing the time of teachers' sessions. It provides features to allocate teaching hours, schedule sessions, and optimize resource utilization.

## Purpose:

The purpose of this documentation is to provide a comprehensive understanding of the time management system for school teachers. It aims to outline the functionalities, features, and requirements of the system to facilitate effective time management for both full-time and contract professors. This system will provide ease for administrations in schools to coordinate teachers' times without the direct need to deal with each teacher's times. The system will automatically reserve the appropriate time for each teacher to give their classes as suits them, taking into account an appropriate educational system for students.

## General Description:

The time management system employs algorithms to intelligently allocate teaching hours based on various parameters such as subject type (literary, scientific, social), weekly time requirements, and section importance. It utilizes a database to store and manage teacher profiles, class schedules, and institutional constraints. The system offers functionalities for scheduling adjustments, conflict resolution, and real-time updates to accommodate unforeseen changes or preferences.

# feasibility study

A feasibility study for the software solution focusing on teacher distribution in schools should assess various aspects to determine its viability. Here are the key areas to consider**:**

## Technical Feasibility:

* The proposed time management system for school teachers is technically feasible,
* leveraging scalable cloud-based architecture,
* intuitive UI design,
* robust security measures,
* seamless integration with existing systems,
* and mobile compatibility while adhering to regulatory standards and cost-effectiveness.

## Operational Feasibility:

* The proposed time management system for school teachers is operationally feasible,
* given its potential to streamline scheduling processes,
* enhance coordination between teachers and administration,
* improve resource allocation,
* and ultimately contribute to more efficient time management practices within educational institutions.

## Financial Feasibility:

* The proposed time management system for school teachers demonstrates financial feasibility through a cost-benefit analysis,
* considering factors such as development,
* deployment,
* maintenance,
* and potential savings from improved efficiency and resource utilization within educational institutions.

## Legal and Compliance Feasibility:

* The system ensures legal and compliance feasibility by adhering to relevant educational regulations,
* data protection laws,
* and labor regulations,
* thus, mitigating legal risks and ensuring alignment with institutional policies and standards.

## Schedule Feasibility:

* The system's development timeline aligns with project goals and deadlines,
* considering factors such as resource availability,
* complexity of features,
* and testing requirements to ensure timely delivery and implementation within educational institutions' operational schedules.

## Market Feasibility:

* The market feasibility for the proposed time management system for school teachers lies in its potential to streamline scheduling processes,
* enhance efficiency,
* and alleviate administrative burdens by automating the allocation of teaching times based on individual preferences and educational requirements.

## Risk Analysis:

* The risk analysis for the proposed time management system for school teachers includes potential challenges such as user adoption resistance,
* technical glitches or system failures,
* data privacy concerns,
* integration issues with existing school systems,
* and the need for ongoing updates and maintenance to ensure optimal performance and usability.

# System Requirements:

## Functional Requirements:

* + 1. User Authentication:

Secure login mechanism for administrators or other stakeholders.

* + 1. Teacher Profile Management:

Ability to add, modify, and delete teacher profiles, including details such as availability and preferences.

* + 1. Session Scheduling:

Functionality to schedule teaching sessions considering teacher availability, subject requirements, and classroom availability. Add to that the type of educational subject (literary, scientific, social), and the time required to be given per week.

* + 1. Conflict Resolution:

Automated detection and resolution of scheduling conflicts to ensure smooth operation. Giving acceptable and logical results, and in solving the deficit, it must give the closest acceptable model.

* + 1. Reporting and Analytics:

Generation of reports and analytics to monitor teacher workload, session distribution, and resource utilization.

## Non-Functional Requirements:

* + 1. Timing Constraints:

The system should generate class schedules within a reasonable time frame to meet school deadlines (from 1s to 1 day at most).

* + 1. Reliability:

The system should be highly reliable, minimizing errors and downtime during schedule generation and access.

* + 1. Response Time:

The system should respond quickly to user actions, ensuring a seamless user experience.

* + 1. Security:

Access to the system must be authorized by the school principal or the system administrator. We must protect the personal information of teachers and businesses.

* + 1. Capacity:

We have 12 academic stages, each stage has at least two sections. The system should efficiently manage and store large volumes of data related to teacher availability, class schedules, and preferences.

* + 1. Usability:

Intuitive user interface with clear navigation and instructional prompts. Accessibility features to accommodate users with diverse needs.

# User stories:

User requirements as user stories for the software solution focused on teacher distribution in schools:

## As a school administrator:

* I want to input teacher availability and preferences
* I want to input subject hours per class
* So that I can efficiently assign teachers to classes based on their schedules and subject preferences.
* I want to generate class schedules for the upcoming semester
* So that I can plan classroom assignments and allocate resources effectively.
* I want to export class schedules in a printable format
* So that I can distribute them to teachers, students, and other stakeholders.
* I want to manually override the automated teacher distribution in specific cases
* So that I can accommodate special requests or handle exceptional circumstances.
* I want to view reports on teacher workload and distribution
* So that I can ensure equitable teacher assignments and optimize resource utilization.

## As a teacher:

* I want to indicate my preferred teaching subjects and grade levels
* I want to input time availability
* So that I can be assigned to classes that align with my expertise and interests.
* I want to receive notifications of any changes to my teaching schedule **(email, phone)**
* So that I can adjust my plans accordingly and avoid conflicts.
* I want to access the class schedule from my mobile device **(email, phone)**
* So that I can easily reference my teaching assignments on the go.

## As a substitute teacher:

* I want to receive notifications of available teaching opportunities
* So that I can quickly accept or decline substitute teaching assignments.

# Functional requirements: Use Cases Description Card:

|  |  |
| --- | --- |
| ID | CLASS\_SCHEDULE\_001 |
| Name | Enter Class Schedule for Each Grade |
| Description | This use case involves the administrator creating a schedule for each grade, detailing the subjects and number of hours for each subject. |
| Pre-condition | 1. The administrator has logged into the software system.  2. Classes from grade 1 to 12 and subjects are already registered in the system. |
| Event flow | 1. The administrator selects the option to create a class schedule.  2. The system prompts the administrator to select a grade.  3. The administrator selects a grade from the available list.  4. The system displays a list of subjects for the selected grade.  5. The administrator enters the number of hours for each subject for the selected grade.  6. The administrator confirms the schedule, and the system saves the data.  7. The system updates the class schedule for the selected grade with the entered information. |
| Extension points | - Integration with teacher assignments to ensure teachers are available for scheduled subjects.  - Automatic notification teachers regarding the class schedule updates. |
| Triggers | Preparation for the upcoming academic year or semester. |
| ID | **SESSION\_STRUCTURE\_002** |
| Name | Define Class Session Structure |
| Description | This use case involves the administrator defining the structure of class sessions, including the number of sessions, break times, and duration of each session, and the day of work per week. |
| Pre-condition | 1. The administrator has logged into the software system.  2. The class schedule for each grade has been created. |
| Event flow | 1. The administrator selects the option to define class session structure.  2. The system prompts the administrator to select a grade.  3. The administrator selects a grade from the available list.  4. The system displays the current session structure (if any) for the selected grade.  5. The administrator defines the number of sessions in a class, the duration of each session, and the duration of breaks between sessions.  6. The administrator confirms the session structure, and the system saves the data.  7. The system updates the class session structure for the selected grade with the entered information. |
| Extension points | - Integration with class schedules to ensure sessions align with subject hours.  - Automatic notification of teachers and students regarding changes in session structure. |
| Triggers | Preparation for the upcoming academic year or semester. |
| ID | **TEACHER\_DIST\_003** |
| Name | Enter the Number of Hours of Each Subject for Each Class |
| Description | This use case involves the administrator entering the number of hours allocated for each subject in each class. |
| Pre-condition | 1. The administrator has logged into the software system.  2. Necessary permissions are granted to the administrator for managing class schedules and subject allocations. |
| Event flow | 1. The administrator selects the option to manage class schedules and subject allocations.  2. The system prompts the administrator to select a class.  3. The administrator inputs the number of hours allotted for each subject in the selected class.  4. The administrator confirms the input, and the system saves the data.  5. The system updates the class schedule with the entered subject hours. |
| Extension points | - Integration with existing curriculum management systems.  - Validation checks to ensure that total subject hours do not exceed the available class time. |
| Triggers | The start of the scheduling process for the upcoming semester. |
| ID | **TEACHER\_DIST\_004** |
| Name | Enter Teacher Availability |
| Description | This use case involves the administrator entering the availability of teachers, specifying their preferred time slots and days. |
| Pre-condition | 1. The administrator has logged into the software system.  2. Teacher profiles are already created in the system. |
| Event flow | 1. The administrator selects the option to manage teacher availability.  2. The system displays a list of available teachers.  3. The administrator selects a teacher and inputs their availability, including preferred time slots and days.  4. The administrator confirms the input, and the system saves the data.  5. The system updates the teacher availability information. |
| Extension points | - Integration with teacher scheduling preferences to optimize teacher assignments.  - Automatic conflict detection for overlapping availability periods. |
| Triggers | Preparation for scheduling the upcoming semester. |
| ID | **TEACHER\_DIST\_005** |
| Name | Enter Teacher Assignments for Each Class |
| Description | This use case involves the administrator assigning teachers to subjects for each class. |
| Pre-condition | 1. The administrator has logged into the software system.  2. Teachers and classes are already registered in the system. |
| Event flow | 1. The administrator selects the option to assign teachers to subjects.  2. The system prompts the administrator to select a class and subject.  3. The administrator selects a teacher from the available list for the selected subject.  4. The administrator confirms the assignment, and the system saves the data.  5. The system updates the teacher assignments for the selected class and subject. |
| Extension points | - Integration with teacher profiles to ensure subject expertise matches assignments.  - Automatic notification of teachers regarding their assigned subjects and classes. |
| Triggers | Preparation for scheduling the upcoming semester. |
| ID | **PRINT\_006** |
| Name | Print Teacher Schedule |
| Description | This use case involves the administrator printing the schedule for a specific teacher. |
| Pre-condition | 1. The administrator has logged into the software system.  2. Teacher schedules are established in the system. |
| Event flow | 1. The administrator selects the option to print a teacher schedule.  2. The system prompts the administrator to select the teacher for whom the schedule will be printed.  3. The administrator selects the desired teacher from the available list.  4. The system generates a printable version of the teacher's schedule.  5. The administrator prints the schedule using the designated printer. |
| Extension points | - Option to select multiple teachers and print their schedules simultaneously.  - Customization options for the printed schedule format (e.g., including/excluding specific details). |
| Triggers | Administrators need to generate physical copies of teacher schedules for various purposes within the school environment. |

# External Interface Requirements:

## User Interface Requirements:

* + 1. Intuitive and user-friendly interface for administrators to input and manage scheduling parameters.
    2. Accessible interface for teachers to view and manage their assigned class times and preferences.

## Hardware Interface Requirements:

* + 1. Compatible with standard computing hardware such as desktops, laptops, and tablets.

## Software Interface Requirements:

* + 1. Integration with existing school management systems for data exchange and synchronization.
    2. Compatibility with popular web browsers for online access.

## Communication Interface Requirements:

* + 1. Support for email notifications and alerts to notify users of scheduling updates or conflicts.

## Other:

* + 1. Performance: Ensure that the software performs efficiently, with fast response times for generating schedules, updating data, and accessing information.
    2. Compliance: Ensure that the software complies with relevant regulations and standards, such as data protection laws and educational standards.
    3. Localization: Support multiple languages and regional settings to accommodate users from diverse backgrounds and locations.

# Definitions and Acronyms:

## Definitions:

### Class Schedule:

A timetable specifying the time, location, and subjects for classes in a school.

### Grade Levels:

Divisions of students based on their academic year or age, typically ranging from grade 1 to grade 12.

### Optimization:

The process of finding the best possible solution among a set of alternatives, often considering constraints and objectives.

### Constraints:

Limitations or rules that must be satisfied when generating schedules, such as teacher availability, classroom capacity, and subject requirements.

### User Roles:

Different categories of users with specific permissions and responsibilities within the software, such as administrators, teachers, students, and parents.

### Integration:

The process of combining different software systems or modules to work together seamlessly, often through standardized interfaces or protocols.

### Accessibility:

The degree to which software can be used by people with disabilities, including those with visual, auditory, motor, or cognitive impairments.

## Acronyms:

|  |  |
| --- | --- |
|  | |
| API: | Application Programming Interface |
| LDAP: | Lightweight Directory Access Protocol |
| SQL: | Structured Query Language |
| HTTPS: | Hypertext Transfer Protocol Secure |
| LDAP: | Lightweight Directory Access Protocol |
| SSO: | Single Sign-On |
| UI: | User Interface |
| UX: | User Experience |
| SRS: | Software Requirements Specification |
| GUI: | Graphical User Interface |

# Choice of SDLC Model:

Given the dynamic nature of software development in educational environments and the need for iterative refinement based on user feedback, the Agile Model would be most suitable. This model allows for continuous collaboration between developers and stakeholders, enabling rapid iterations and incremental enhancements to meet evolving requirements effectively. Agile's flexibility and adaptability align well with the iterative nature of educational processes and the need for frequent updates and improvements in the time management system.

Adapt the Agile Software Development Life Cycle (SDLC) for the development of a software solution for teacher distribution in schools:

## Project Initiation:

* Formulate the project vision and goals, understanding the need for an efficient teacher distribution system in schools.
* Identify key stakeholders, including school administrators, teachers, and potentially students.
* Establish the initial product backlog, listing high-level features and functionalities based on user needs and requirements (open then closed interview).

## Sprint 0:

* Conduct initial planning and setup activities, including environment setup, tool selection, and team onboarding.
* Define the architecture and technology stack for the software solution.
* Set up communication channels and collaboration tools for the development team.

## Sprint Planning:

* Select a subset of features from the product backlog for the first sprint.
* Break down selected features into smaller tasks with clear acceptance criteria.
* Estimate the effort required for each task and commit to completing them within the sprint.

## Sprint Execution:

* Develop the software incrementally, focusing on implementing the tasks committed to in the sprint plan.
* Collaborate closely with stakeholders to clarify requirements and address any questions or concerns.
* Conduct weekly stand-up meetings to discuss progress, identify any impediments, and plan work for the day.

## Sprint Review:

* Demonstrate the completed features to stakeholders, including school administrators and teachers.
* Gather feedback on the implemented features and incorporate any necessary adjustments.
* Review the sprint process and identify lessons learned for continuous improvement.

## Sprint Retrospective:

* Reflect on the sprint process and team dynamics, identifying what went well and areas for improvement.
* Discuss any process improvements or changes to be implemented in the next sprint.
* Update the product backlog based on feedback and changing priorities.

## Repeat:

* Continue iterating through subsequent sprints, each focused on delivering incremental value to the users.
* Regularly review and prioritize the product backlog based on stakeholder feedback and evolving requirements.
* Adapt the development approach as needed based on project progress and feedback received during sprint reviews and retrospectives.

The choice of the Agile model for developing the time management system for schools is based on several factors that align with the specific needs and context of the project:

* Iterative Development: Agile methodologies, such as Scrum or Kanban, emphasize iterative development cycles. In the context of educational institutions, where requirements might evolve or change due to varying academic schedules, teaching preferences, or administrative policies, an iterative approach allows for flexibility and adaptability. This means that the system can be developed incrementally, with features being added or modified in response to ongoing feedback from teachers, administrators, and other stakeholders.
* Stakeholder Collaboration: Agile methodologies prioritize close collaboration between developers and stakeholders throughout the development process. In the case of a time management system for schools, involving teachers, administrators, and other relevant parties in the development process ensures that the system meets their specific needs and addresses any unique challenges or requirements they may have. Regular feedback loops, such as sprint reviews or daily stand-up meetings, facilitate communication and alignment between the development team and the end users.
* Responsive to Change: Agile methodologies are designed to embrace change, recognizing that requirements and priorities may evolve over time. In the context of an educational environment, where factors such as class schedules, teacher availability, and curriculum requirements can change frequently, the ability to respond quickly to these changes is crucial. Agile practices such as continuous integration, frequent testing, and adaptive planning allow the development team to incorporate changes efficiently and effectively without disrupting the overall development process.
* Early Delivery of Value: Agile methodologies emphasize delivering working software in small, incremental releases. This approach enables stakeholders to see tangible progress and derive value from the system early in the development process. For a time, management system for schools, this means that essential features can be prioritized and delivered incrementally, allowing teachers and administrators to start benefiting from the system sooner rather than later.

Overall, the Agile model is well-suited for the development of the time management system for schools due to its iterative nature, emphasis on stakeholder collaboration, responsiveness to change, and focus on delivering value early and continuously. These characteristics align closely with the dynamic and evolving nature of educational environments, making Agile an appropriate choice for this project.