

Software Engineering

Software requirement specification (SRS)



March 11, 2024

beirut.tech

Beirut- Airport Road



|  |  |
| --- | --- |
| **Project name:** | Teachers' class schedule |
| **Date:** | 11/3/2024 |
| **Version:** | 2.3 |
| **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: 3](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981428)

[Product Scope: 3](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981429)

[General Description: 4](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981433)

[feasibility study 5](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981434)

[System Requirements: 7](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981442)

[Functional Requirements: 7](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981443)

[Non-Functional Requirements: 8](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981449)

[User stories: 9](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981456)

[Functional requirements: Use Cases Description Card: 10](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981460)

[External Interface Requirements: 13](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981461)

[Definitions and Acronyms: 14](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981467)

[Choice of SDLC Model: 15](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981477)

[Diagrams: 17](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981485)

[Class Diagram: 17](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981486)

[Use Case Diagram 18](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981487)

[Activity Diagram: 19](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981488)

[Data Flow: 19](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981490)

[Data Flow Level 0: 20](file:///C:\Users\mhmdjawhari\Desktop\Software%20Engineering-B\SE_project\phase%205.2.docx#_Toc163981491)

# 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:

* Technical Resources: We have the necessary technology and equipment.
* Expertise: We have the right technical skills.
* Technical Risks: No potential technical issues.
* Integration: The system can be integrated with existing systems or processes
* Scalability: The system can scale to meet future requirements.

## Operational Feasibility:

* User Acceptability: The system is acceptable to its users and meet their requirements.
* Management Support: There is support from management for the system and they committed to its success.
* Integration: The system can be integrated with existing systems without too much disruption.
* Resources: There are sufficient resources (time, budget, personnel) available to implement and maintain the system.
* Expertise: There is enough expertise available to design, implement, and maintain the system.
* The proposed software system will integrate with existing systems and procedures.

## Financial Feasibility:

* The system is financially feasible, its initial costs and ongoing operational costs are less than the budget.
* projected revenues are more than the budget.
* The customer will pay cash.

## Legal and Compliance Feasibility:

* Identify Relevant Laws: Laws and regulations apply to the Teachers' class schedule system. This includes local, state, federal, and international laws depending on the scope of the system.
* Consult Legal Experts: Legal professionals who specialize in the relevant areas of law says that the system is legal.
* Conduct a Risk Assessment: No potential legal risks associated with the system.
* Compliance Check: The system complies with all the identified legal requirements.
* Documentation: We keep thorough documentation of all compliance efforts and legal consultations.

## Schedule Feasibility:

* The system's development timeline aligns with project goals and deadlines.
* Resources are available.
* Features are not complex.
* We have testing requirements to ensure timely delivery and implementation within educational institutions' operational schedules.

## Market Feasibility:

* Market Research: The system is in demand; it fits customer needs and has competition in the target market (has no effective alternative).
* Financial Analysis: The cost of development and operation is not comparable to the potential revenue.
* Regulatory Compliance: The system meets all legal and regulatory requirements for the market.
* Risk Assessment: No serious potential risks.

## Risk Analysis:

* User adoption resistance.
* Technical glitches or system failures.
* Data privacy concerns.
* Integration issues with existing school systems.
* 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:

* + - Timing Constraints:

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

* + - Reliability:

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

* + - Response Time:

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

* + - 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 institutions.

* + - 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.

* + - 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 Head of Department:

1. As a head of department, I want to log in to manage the system so that I can add classes and open new sections if needed.
2. As a Head of Department, if a class is full, I want to open a new section for it.
3. As a head of department, I want to add teachers to classes so that I can ensure all classes are properly staffed.
4. As a Head of Department, if the schedule is finalized, I want to make the final decision. If accepted, the schedule is published and printed; otherwise, I want to request the principal to redo it.

## As a principal:

1. As a principal, I want to login to manage the system so that I can control book times and add teachers.
2. As a principal, I want to manage the software to open/close time slots for booking so that I can control the scheduling process effectively.
3. As a principal, I want to view schedules of teachers, classes, and departments so that I can monitor and manage school resources efficiently.

## As a teacher:

1. As a teacher, during the designated time for adding or modifying availability, I want to add my available hours.
2. As a teacher, I want to view the schedule after it has been finalized by the system so that I can plan accordingly.
3. As a teacher, I want to receive notifications of any changes to my teaching schedule via email or phone so that I can adjust my plans accordingly and avoid conflicts.

# Functional requirements: Use Cases Description Card:

|  |  |
| --- | --- |
| ID | HEAD\_LOGIN\_001 |
| Name | Log In to Manage the System |
| Description | Head of Department logs into the system to manage classes, open new sections, and oversee scheduling. |
| Pre-condition | None |
| Event flow | The head of department enters their username and password.  The system authenticates the credentials provided by the head of department.  Upon successful authentication, the head of department gains access to management functions. |
| Extension points | None |
| Triggers | The need to manage the system and add classes or open new sections. |
| ID | **HEAD\_OPEN\_NEW\_SECTION\_002** |
| Name | Open New Section for Full Class |
| Description | Head of Department initiates the process of opening a new section for a class that has reached its maximum capacity. |
| Pre-condition | The head of department has logged into the system.  The class has reached its maximum capacity. |
| Event flow | The head of department identifies the class that has reached full capacity.  The head of department selects the option to open a new section for the class.  The system creates a new section for the class with the same curriculum. |
| Extension points | None |
| Triggers | A class reaches its maximum capacity, requiring the opening of a new section. |
| ID | **HEAD\_FINAL\_DECISION\_003** |
| Name | Make Final Decision on Schedule |
| Description | Head of Department reviews and decides whether to accept or request revisions to the finalized schedule. |
| Pre-condition | The head of department has logged into the system. |
| Event flow | The head of department reviews the finalized schedule.  The head of department decides whether to accept the schedule or request revisions. |
| Extension points | None |
| Triggers | Completion of the scheduling process. |
| ID | **HEAD\_ADD\_TEACHERS\_004** |
| Name | Add Teachers to Classes |
| Description | Head of Department assigns teachers to classes to ensure proper staffing. |
| Pre-condition | The head of department has logged into the system. |
| Event flow | The head of department selects the option to add teachers to classes.  The head of department assigns teachers to the respective classes |
| Extension points | None |
| Triggers | The need to staff classes appropriately. |
| ID | **PRINCIPAL\_LOGIN\_005** |
| Name | Log In to Manage the System (Principal) |
| Description | Principal logs into the system to control book times and add teachers. |
| Pre-condition | None |
| Event flow | The principal enters their username and password.  The system authenticates the credentials provided by the principal.  Upon successful authentication, the principal gains access to management functions. |
| Extension points | None |
| Triggers | The need to manage the system and control book times or add teachers. |
| ID | **PRINCIPAL\_VIEW\_SCHEDULES\_006** |
| Name | Principal Views Schedules |
| Description | Principal views schedules of teachers, classes, and departments to oversee school operations. |
| Pre-condition | Principal has logged into the system. |
| Event flow | Principal navigates to the schedule viewing section.  Principal selects the option to view schedules.  System displays schedules of teachers, classes, and departments. |
| Extension points | None |
| Triggers | Need to oversee school operations. |
| ID | **PRINCIPAL\_OPEN\_CLOSE\_TIMESLOTS\_007** |
| Name | Open/Close Time Slots for Booking |
| Description | Principal opens or closes time slots for booking to manage the scheduling process effectively. |
| Pre-condition | The principal has logged into the system. |
| Event flow | The principal accesses the time slot management section.  The principal selects the option to open or close time slots.  The system updates the booking availability accordingly. |
| Extension points | The head of department has completed the task of adding teachers to classes. |
| Triggers | The need to manage the scheduling process effectively. |
| ID | **TEACHER\_ADD\_MODIFY\_AVAILABILITY\_008** |
| Name | Teacher Adds/Modifies Availability |
| Description | Teacher adds or modifies availability within the given timeframe to ensure schedule alignment with personal preferences. |
| Pre-condition | Teacher schedules are established in the system.  The booking availability is open.  The teacher's name is listed on the class. |
| Event flow | The teacher selects the option to add or modify availability.  The teacher specifies their available times/days.  The system saves the availability information provided by the teacher. |
| Extension points | None |
| Triggers | The need to align the schedule with personal preferences. |
| ID | **TEACHER\_VIEW\_FINAL\_SCHEDULE\_009** |
| Name | View Finalized Schedule (Teacher) |
| Description | Teacher views finalized schedule to plan accordingly. |
| Pre-condition | The schedule is completed by the system, and the head of department approves it. |
| Event flow | The system finalizes the schedule.  The system notifies the teacher that the schedule is available for viewing.  The teacher accesses the finalized schedule. |
| Extension points | None |
| Triggers | Completion of the scheduling process. |

# External Interface Requirements:

## User Interface Requirements:

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

## Hardware Interface Requirements:

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

## Software Interface Requirements:

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

## Communication Interface Requirements:

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

## Other:

* + - Performance: Ensure that the software performs efficiently, with fast response times for generating schedules, updating data, and accessing information.
    - Compliance: Ensure that the software complies with relevant regulations and standards, such as data protection laws and educational standards.
    - 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:

|  |  |
| --- | --- |
|  | |
| UI: | User Interface |
| UX: | User Experience |
| SRS: | Software Requirements Specification |
| GUI: | Graphical User Interface |

# Choice of SDLC Model:

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.

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.

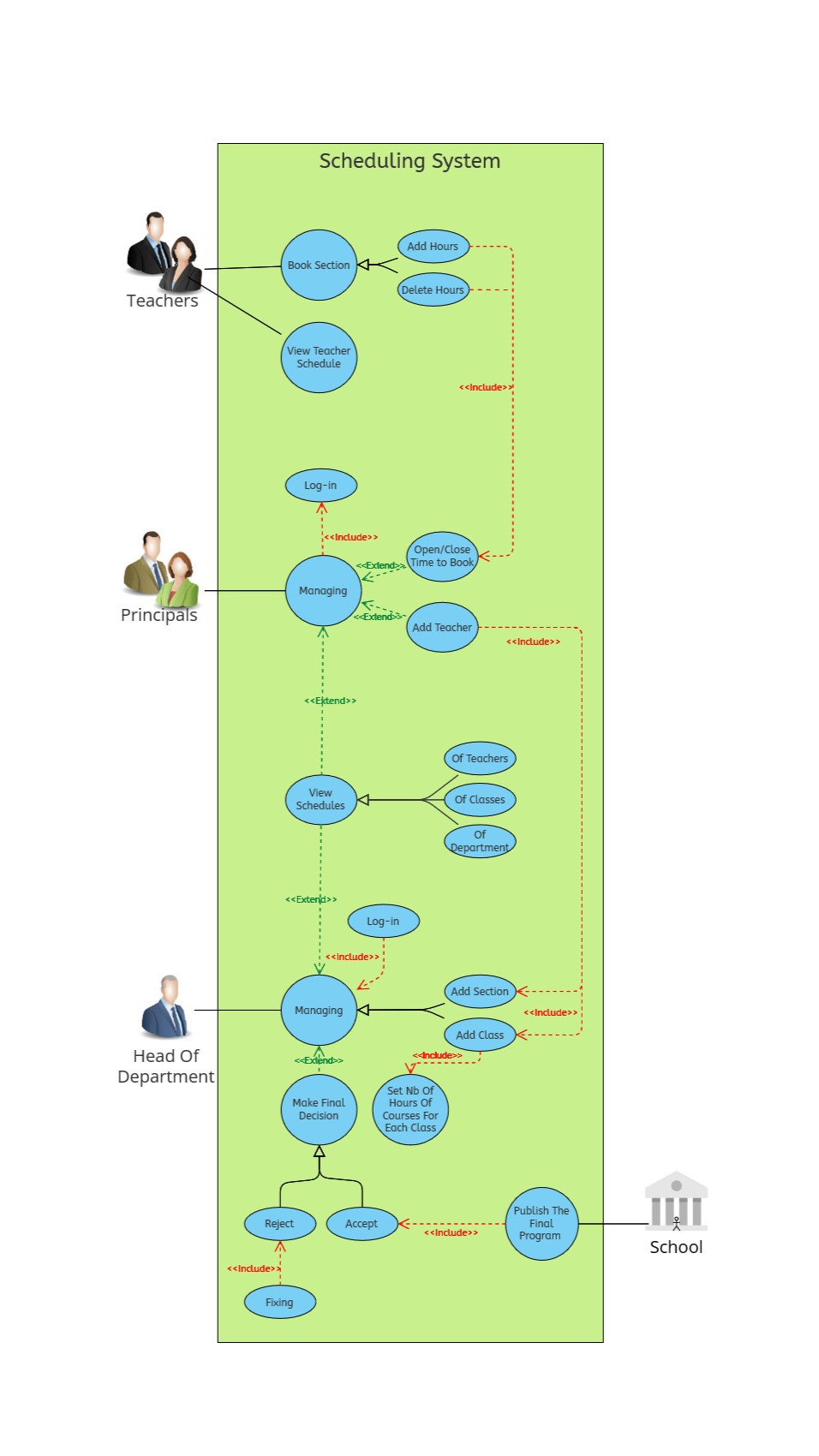
## Repeat:

* 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.

# Diagrams:

## Class Diagram:

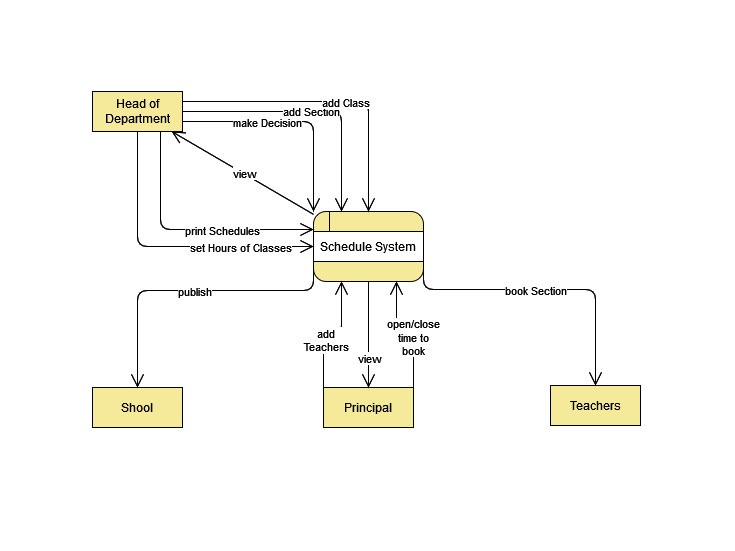
## Use Case Diagram



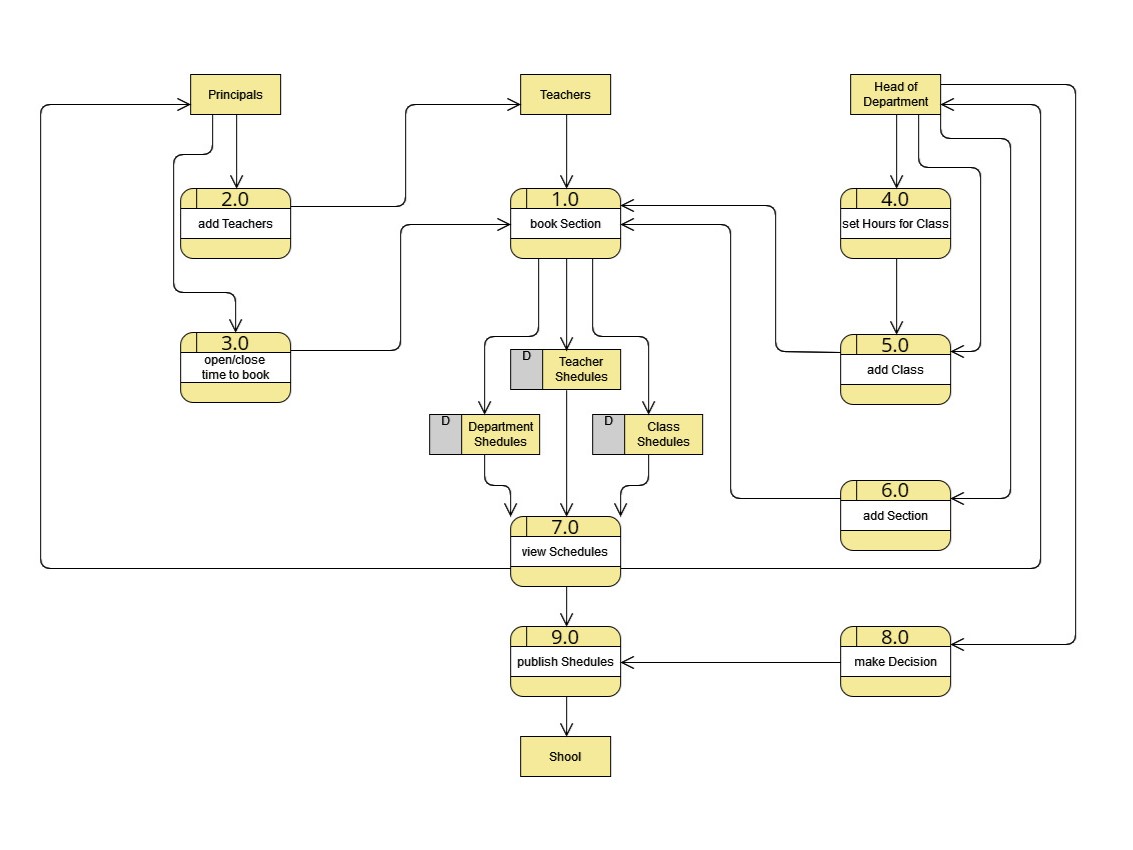
## Activity Diagram:

## 

## Data Flow:



## Data Flow Level 0:

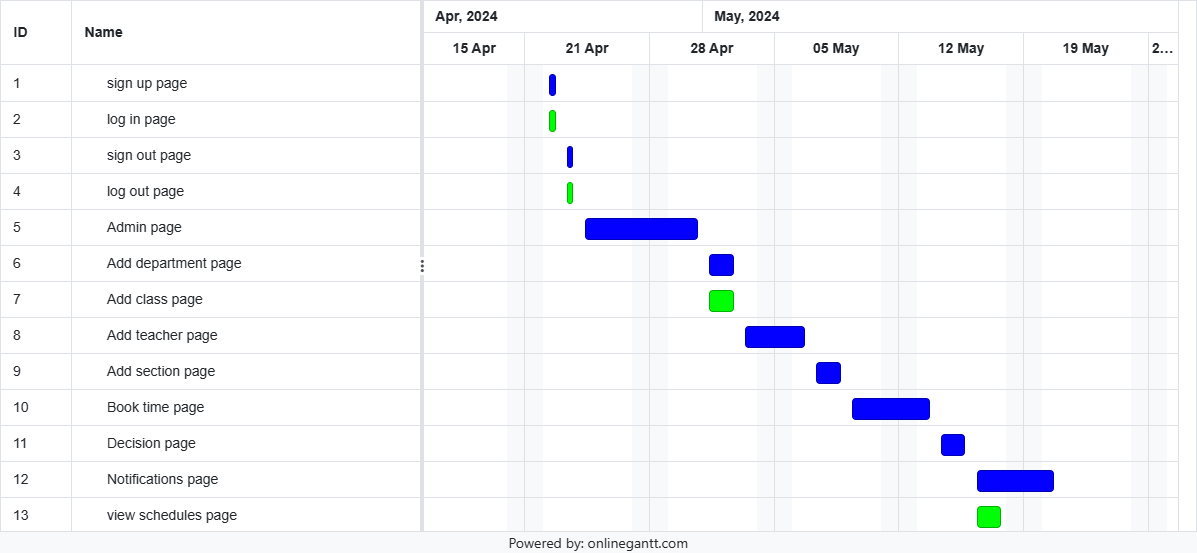


# Charts: Pert Chart:

## 

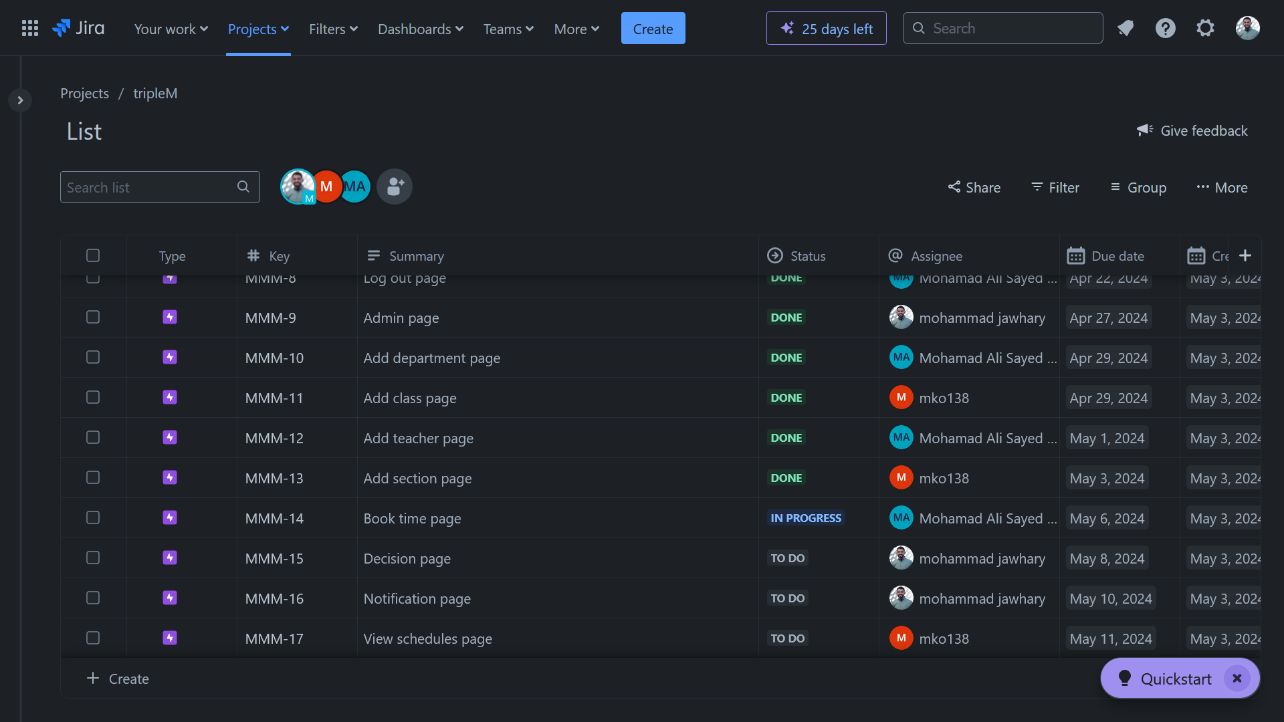
## Gant Chart:

## 

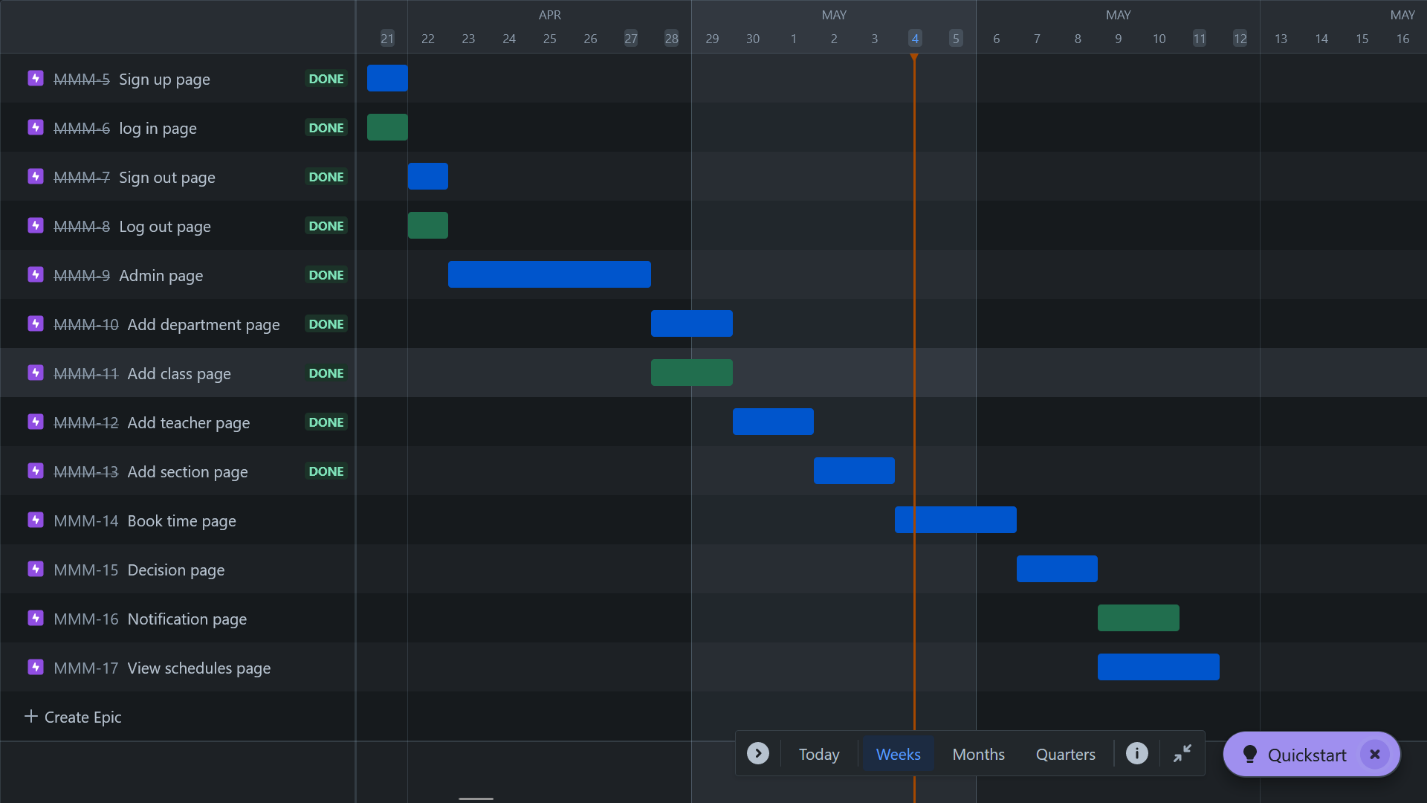


# Practice on Jira:

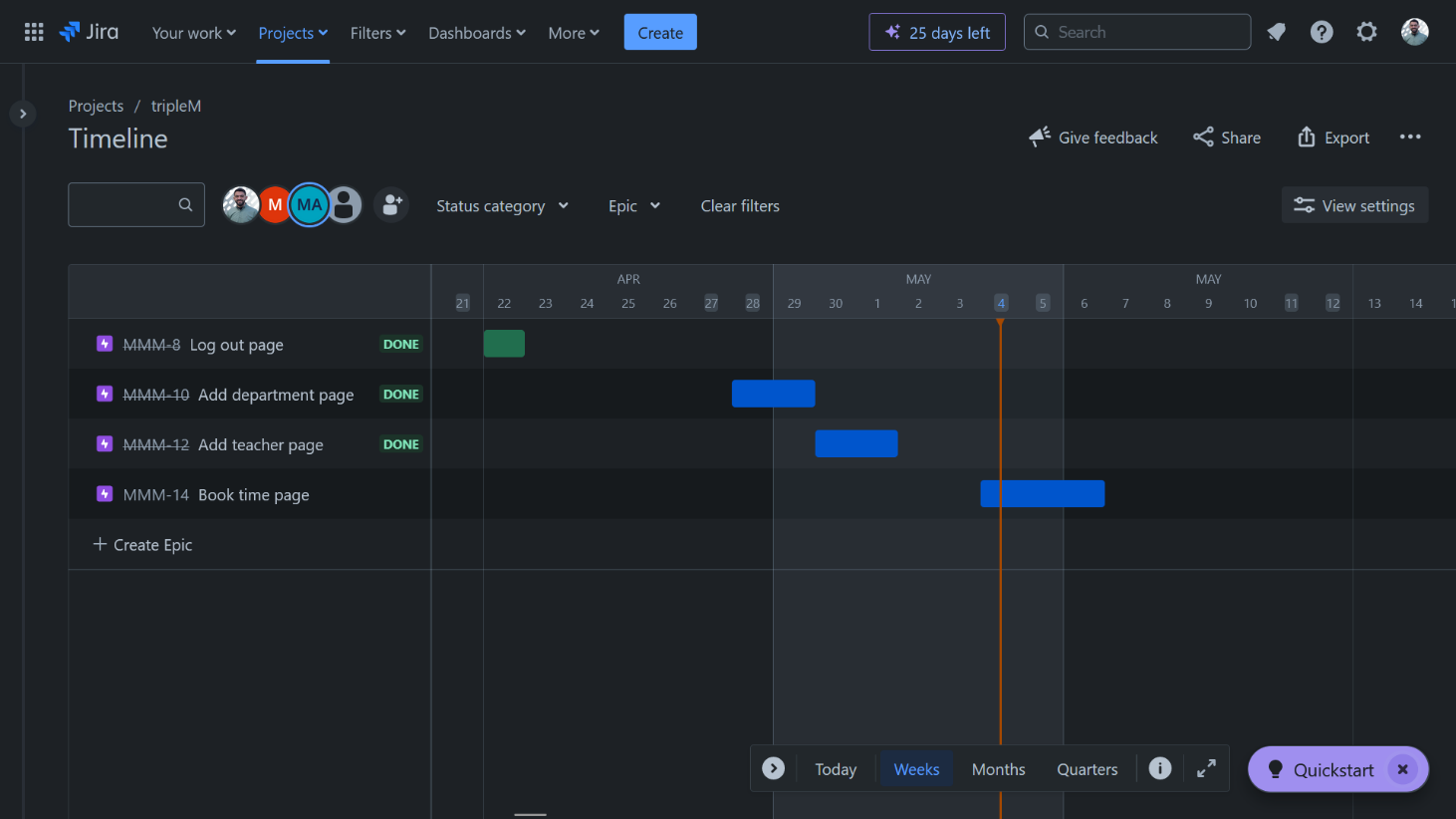
## Our list of tasks:

****

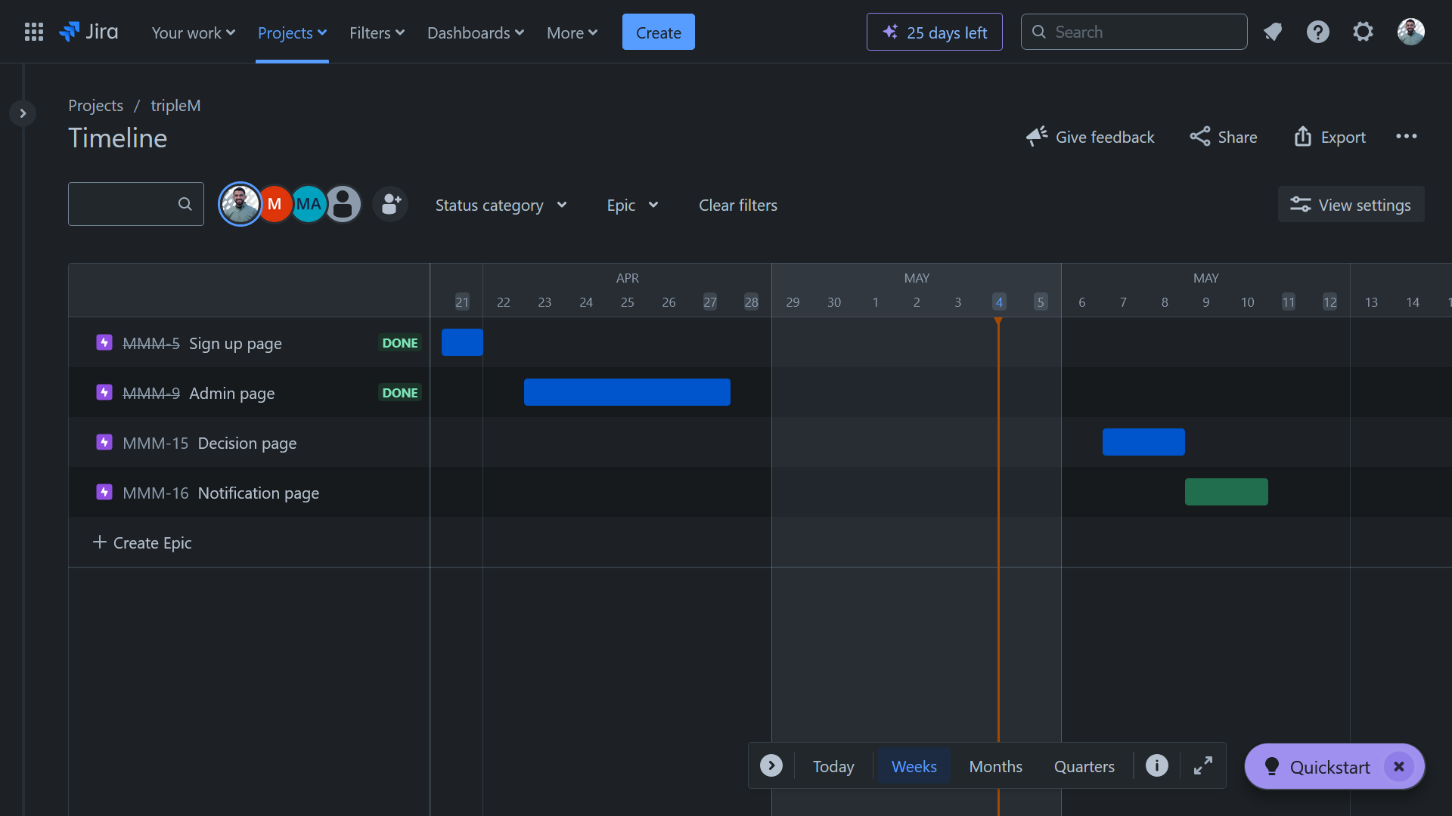
## Timeline over all:

****

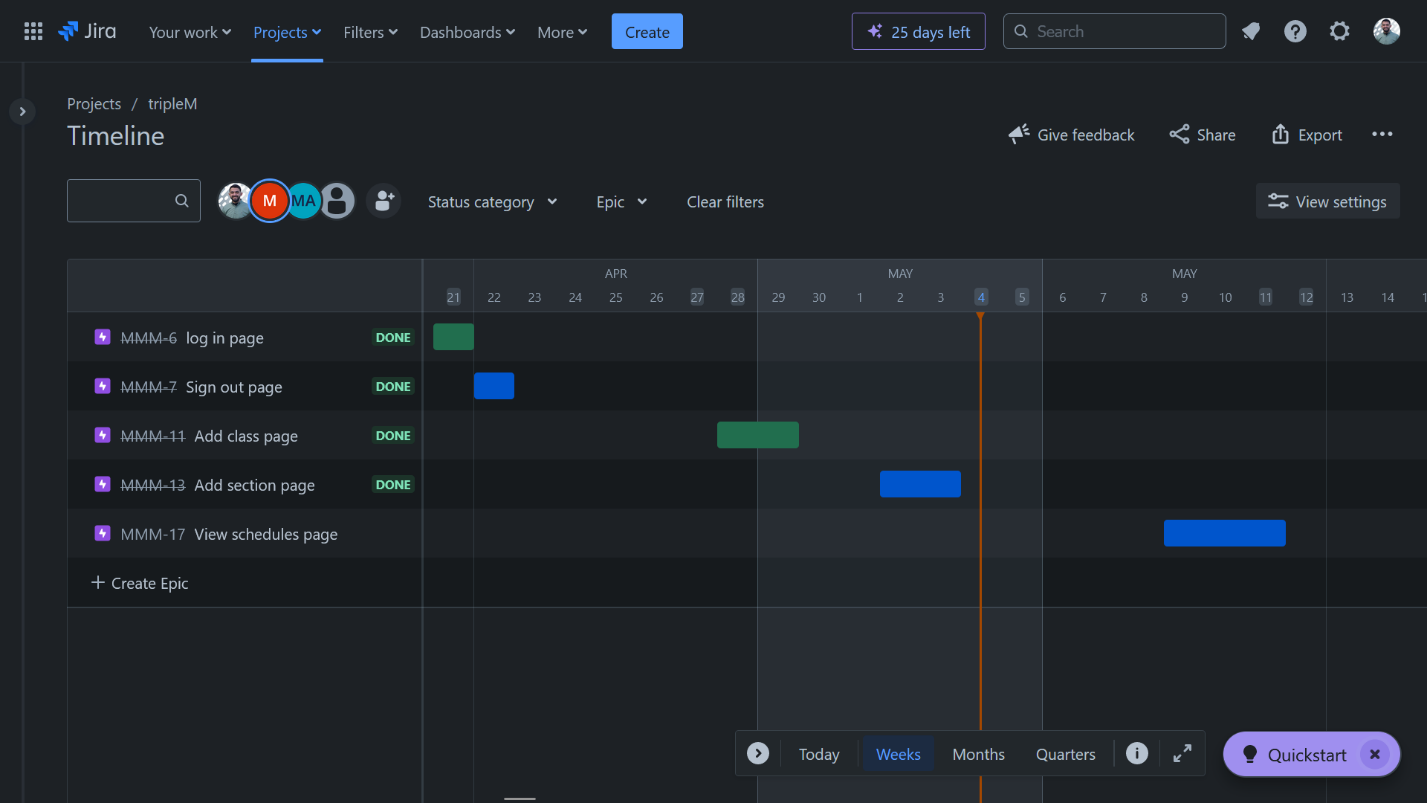
## Timeline for Mohamad Ali Sayed Ahmad:

****

## Timeline for Mohamad Al Jawhari:

****

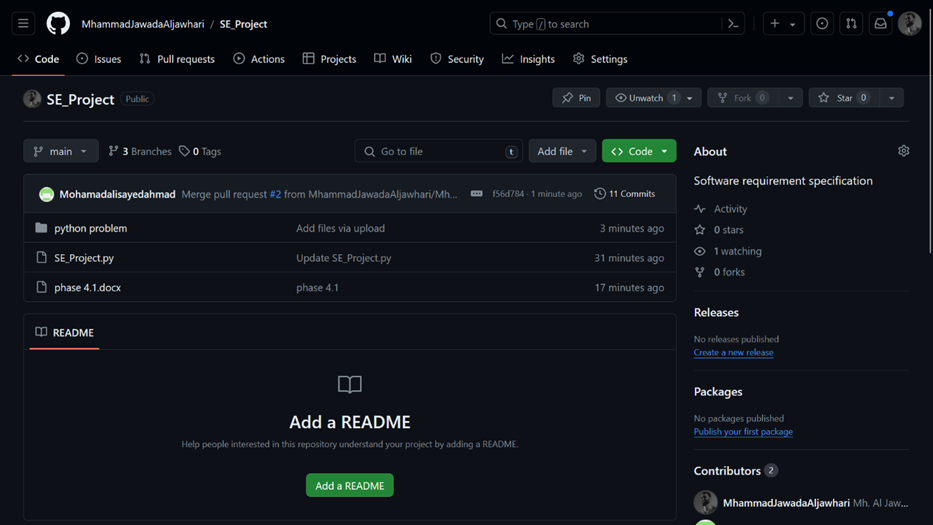
## Timeline for Mohamad Obayd:

****

# Practice on GitHub:

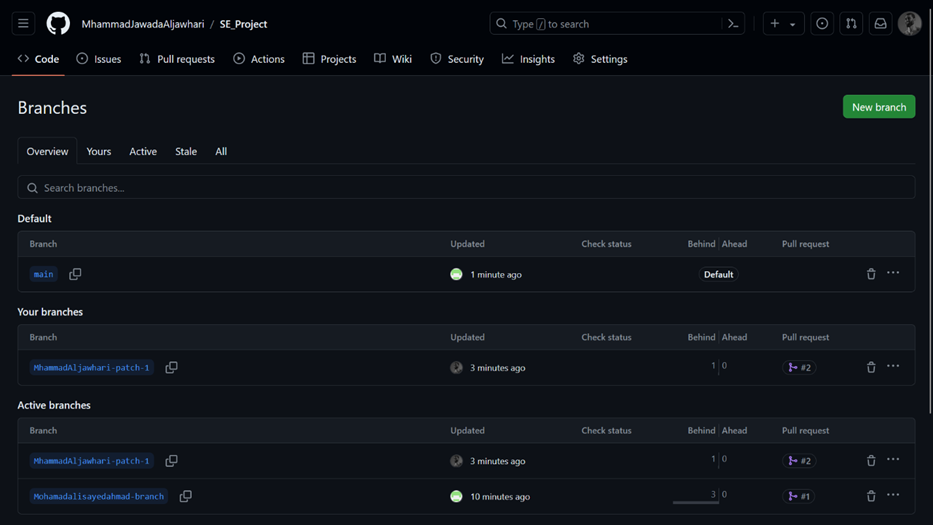


## Create a repository named SE\_Project by MhammadJawadaAljawhari:





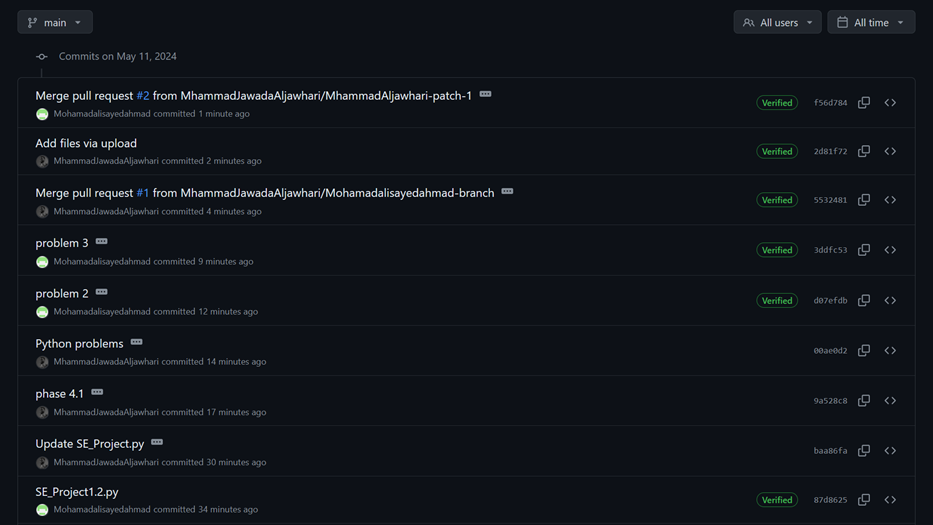
## Create some Branches:

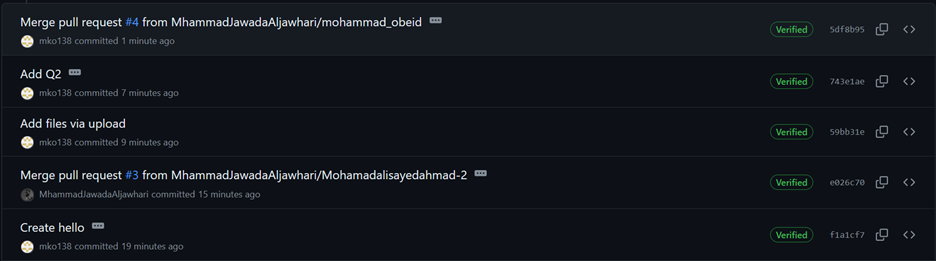


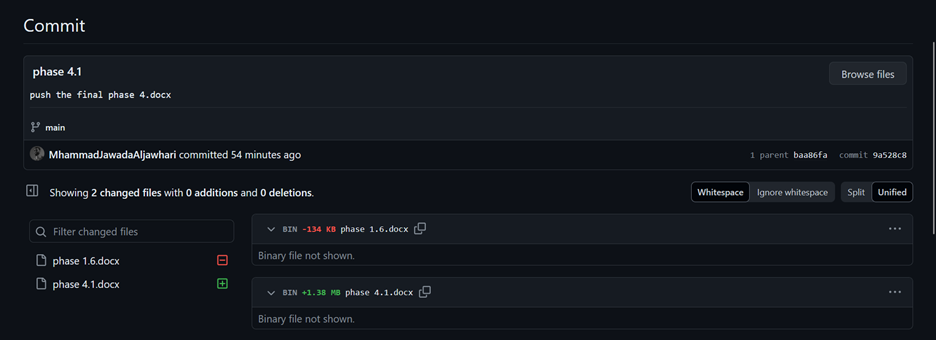
One mean branch (Default one)

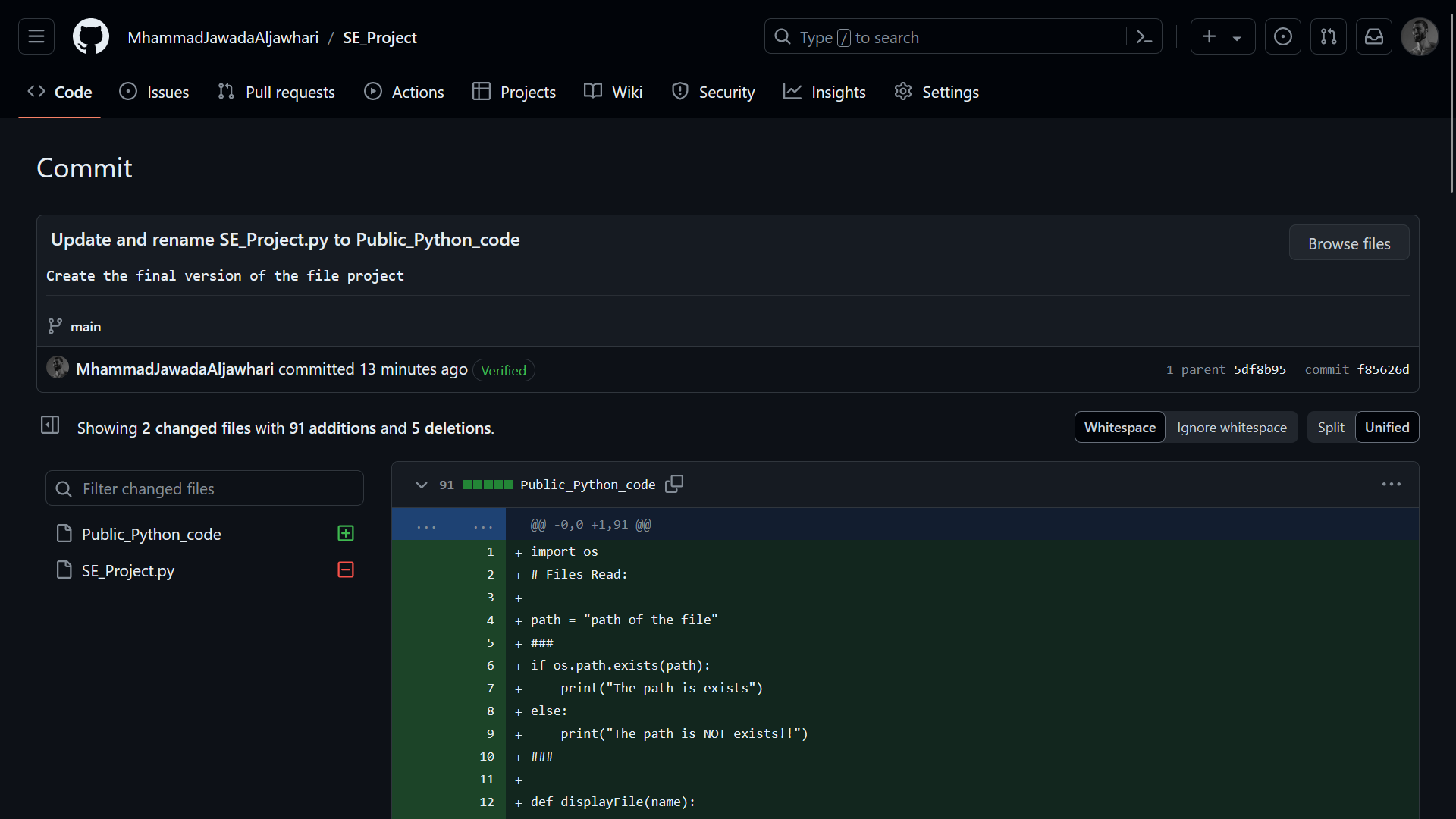
Tow active branch, one for Mohamad Ali Sayed Ahmad (Mohamadalisayedahmad-branch) and the other one for Mohamad Al Jawhari (MhammadAljawhari-patch-1)

## Using the Branches we practice on merge and how to pull & push:

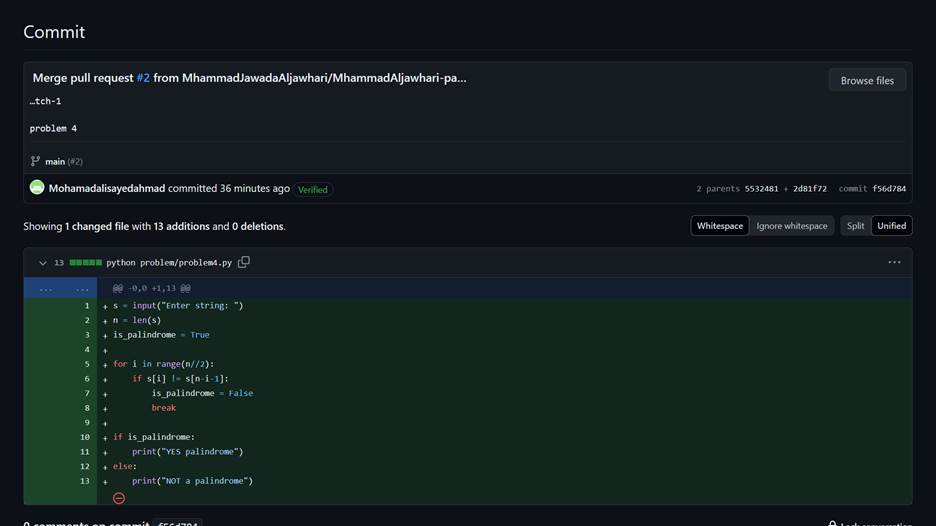








And here we can see the version of some files, and we can go back to old version.



We have been practicing branches, merges, push, and pull operations. Here's a glimpse of our progress in the attached screenshot. It's been a valuable learning experience for us, emphasizing the importance of version control and teamwork.