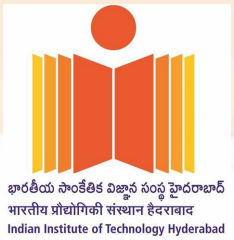
SAP SRS Documentation



SAP: Seat Allotment Portal

Software Requirements Specification

Course: Software Engineering

Course code: CS4443

Faculty Guide: *Dr. MV Panduranga Rao*

Student Developers:

Bhavanam Sujeeth Kumar Reddy (ES19BTECH11022)

Krishn Vishwas Kher (CS19B23P000001)

Mukkavalli Bharat Chandra (ES19BTECH11015)

Sree Prathyush Chinta (CS19BTECH11043)

Seat Allotment Portal tailored for managing CoAP/GATE admisssions @ IITH.

Contents

Chapter 1

Introduction

1.1 Purpose

Our system is designed to make it more convenient for the members of the student short-listing committee dealing with the applications for PG programmes offered at the institute by allocating the seats to the students automatically with 100 % accuracy. This document provides an overview of the characteristics of our system, giving the client a lucid picture of the product we offer alongside serving as a roadmap for our developers.

1.2 Scope

Below enumerated are the aspects of our system that fall within the scope of our system as well as those that fall out of it.

1.2.1 In scope

- Performing the seat allocation process across various rounds of CoAP procedure.
- Sorting the users based on priorities as well as raising exception errors when ties occur for various candidates.
- Modification of seat matrix at any point during the CoAP process.
- Providing statistics of seat allotment after the entire CoAP process.
- User authentication.

1.2.2 Out of scope

- Uploading the seat allotment output to the CoAP portal.
- Downloading the CoAP response from the portal.
- Verification of user details before sorting.

1.3 Definitions, Acronyms, and Abbreviations

Documentation of computerese and associated meanings that will be used frequently in this document and in subsequent software documents for this project.

1.3.1 Abbreviations & Acronyms

- **SAP**: Seat Allotment Portal.
- **CoAP**: Common Offer Acceptance Portal.
- **PG**: Post Graduate.

1.3.2 Definitions

- **Sorting**: Ordering the candidates based on the priorities specified by the user.
- **Seat Matrix**: The seats count division across various specialisations and subsequent divisions based on category (under each specialisation), as mandated by the user.

1.4 References

1. **Appendix A**: Sorting procedure used by the portal. (5.1)
2. **Appendix B**: Seat Matrix declaration and modification. (5.2)
3. **Appendix C**: User Interface. (5.3)

1.5 Overview

- **Section** 2: Provides a comprehensive overview of the software, including the expected level of proficiency for users, general constraints that must be considered during software development, as well as the underlying assumptions and dependencies.
- **Section** 3: We will delve into the specific requirements that the software must fulfill, with functional requirements presented through various use cases. Additionally, performance requirements and design constraints will also be detailed in this section.
- **Section** 4: Here we will shed light on potential future extensions for the system, which will be of great interest to those who plan to use the software in the long term.
- **Section** 5: Describes few crucial processes that can occur during the seat allocation procedure.

Chapter 2

Overall description

2.1 Product Perspective

The "Seat Allotment Portal" software is aimed for IITH faculty to sort the candidates based on a sorting criteria efficiently and to reduce the manual component involved with the traditional method significantly. SAP should be an easy-to-use, highly reliable software capable of offering a lot of flexibility to the user(s). SAP is intended to be a stand-alone software, independent of the availability of other software. SAP is platform independent, i.e, it should run on Linux, UNIX and Windows based platforms.

2.2 Product Functions

Class of use cases	Use Cases	Description of use cases
Use case related to Installation	Installation	Creates and initializes working files.
Use cases related to system authorization	Registration	Registers new users into SAP.
	Login	Login to SAP
	Change password	Change SAP password
Use cases related to project	Create Project	Create new SAP project
	Rename Project	Rename existing SAP project
	Delete Project	Delete existing SAP project
	Clone Project	Clone existing SAP project
Use cases related to uploading data	Uploading Seat Matrix	Upload seat matrix in round 1
	Uploading the Candidate Data	Uploading the candidate data in round 1
	Uploading the CoAP response	Uploading the CoAP response in subsequent rounds
Use cases related to result retrieval	Sort Functionality	Sorts the candidate data in round 1
	Get Offers	Get the student offers in every round
Other cases	Update seat matrix	Update the seat matrix in subsequent rounds

2.3 Product Planning

Agile methodology will be adopted to ensure the maximum client satisfaction as well as continuous delivery of working code after every few sprints. Code practices that facilitate continuous feedback from the users will be always strived for. This may include changing certain details in the present

SRS document, but the barebone structure will remain invariant.

2.4 User Characteristics

- The intended user is a person empowered by IITH with the academic powers authorized to shortlist students based on various criteria e.g. faculty member of IITH.
- The user should be familiar with the rules needed for shortlisting.
- Familiarly with basic handling of Microsoft Excel/Google sheets is desirable.

2.5 Principal Actors

- `User`.
- `System`.

2.6 General Constraints

1. Should output results with 100% accuracy.
2. It should be efficiently scalable in the number of students being shortlisted.
3. Tool/application should be OS-independent.
4. Requires Internet connection for it to work fully.

2.7 Assumptions & Dependencies

1. Certain features require Internet connectivity.
2. Takes input data only in the form of Microsoft Excel /Google sheets/csv files.
3. 100% accuracy on results guaranteed only when data is 100% clean.
4. The user uploads the correct .csv file containing the candidates data in Round 1.

Chapter 3

Specific Requirements

3.1 Functional Requirements

The functional requirements are described via various use cases.

3.1.1 Installation

- **Use Case** 1 *Installation of SAP*
  - **Primary Actor**: User.
  - **Precondition**: Internet connection available.
  - **Main scenario**:
    1. The user installs the SAP tool from the IITH intranet.
    2. System asks the user for the home directory in which all the working files will be created.
  - **Alternate Scenario**:
    1. Network failure - Installation aborted.

3.1.2 System Authorization

- **Use Case** 2 *Registration on SAP*
  - **Primary Actor**: User.
  - **Precondition**: User must have installed SAP.
  - **Main scenario**:
    1. User gives the IITH email address.
    2. User enters the desired password.
    3. User re-enters the password.
  - **Alternate Scenario**:
    1. User tries to register multiple times using the same IITH email address yet the user fails to do so. A message saying "Account already exists!" is generated.
- **Use Case** 3 *Login*
  - **Primary Actor**: User.
  - **Precondition**: User must have registered on SAP .
  - **Main scenario**:
    1. Start the application. User prompted for login and password.
    2. User gives the login and password.
    3. System does authentication.
    4. Home page is displayed.
  - **Alternate Scenario**:
    1. * Authorization fails.
       * Prompt the user that he typed the wrong password.
       * Allow him to re-enter the password.
    2. * User logins through multiple devices.
       * Log out from the current session.
       * Warns the user to login only through a single device.
- **Use Case** 4 *Change Password*
  - **Primary Actor**: User.
  - **Precondition**: User logged in.
  - **Main scenario**:
    1. User goes to the profile settings page to change the password.
    2. User is prompted for old password, new password and confirm new password.
    3. User gives the old password, new password and confirms the new password.
    4. System does authentication.
    5. New password is registered with the system.
  - **Alternate Scenario**:
    1. * Authorization fails.
       * Prompt the user that he typed the wrong password.
       * Allow him to re-enter the password.
    2. * New password and the confirmed new password do not match.
       * Allow him to re-enter the attributes.

3.1.3 Project

- **Use Case** 5 *Create Project*
  - **Primary Actor**: User.
  - **Precondition**: User logged in.
  - **Main scenario**:
    1. User initiates the “Create Project” functionality.
    2. System asks the user for the project name.
    3. User enters the project name.
    4. An empty project is created.
  - **Alternate Scenario**:
    1. * Project with the same name exists.
       * System asks the user for a different name.
       * User enters a different name.
       * Empty project gets created.
- **Use Case** 6 *Rename Project*
  - **Primary Actor**: User.
  - **Precondition**: User should have an existing project
  - **Main scenario**:
    1. User initiates the “rename project” functionality.
    2. System asks for the project to be renamed and the new name.
    3. User enters the new name.
    4. Project is renamed.
  - **Alternate Scenario**:
    1. * Project with the same name exists.
       * System asks the user for a different name.
       * User enters a different name.
       * Project gets renamed.
- **Use Case** 7 *Delete Project*
  - **Primary Actor**: User.
  - **Precondition**: User should have an existing project
  - **Main scenario**:
    1. User selects a project.
    2. User selects the “Delete Project” functionality for the selected project.
    3. The system deletes the project.
  - **Alternate Scenario**:
    1. None

- **Use Case** 8 *Clone Project*

  - **Primary Actor**: User.
  - **Precondition**: User should have an existing project
  - **Main scenario**:
    1. User selects a project.
    2. User selects the "Clone Project" functionality for the selected project.
    3. System creates a duplicate project with a system generated name.
  - **Alternate Scenario**:
    1. None

3.1.4 Data uploads

- **Use Case** 9 *Uploading Seat Matrix*

  - **Primary Actor**: User.
  - **Precondition**: User has created a new project and has selected Round 1.
  - **Main scenario**:
    1. User either uploads the csv file containing the seat matrix or manually fills in the seat matrix in SAP.
    2. The seat matrix has been successfully uploaded to SAP.
  - **Alternate Scenario**:
    1. ∗ If a user violates the seat matrix format in SAP in the uploaded csv file.
       ∗ System generates a warning message.
       ∗ The user is prompted to upload the file again.
- **Use Case** 10 *Uploading the Candidate Data in Round 1*

  - **Primary Actor**: User.
  - **Precondition**: User has uploaded the seat matrix
  - **Main scenario**:
    1. User uploads the candidate data (csv file) on SAP.
    2. The upload is successful.
    3. Preview of the uploaded csv file is shown on the screen.
  - **Alternate Scenario**:
    1. None

- **Use Case** 11 *Upload the CoAP response*

  - **Primary Actor**: User.
  - **Precondition**: User is accessing a subsequent round (not Round 1)
  - **Main scenario**:
    1. User uploads the CoAP response of the previous round and generates a new list of offers for that particular round.
  - **Alternate Scenario**:
    1. ∗ User should upload the correct CoAP response file. A prompt will be generated if an incorrect file is uploaded.

3.1.5 Result retrieval

- **Use Case** 12 *Sort Candidate Data*

  - **Primary Actor**: User.
  - **Precondition**: User has uploaded the candidate data in Round 1.
  - **Main scenario**:
    1. User sets the sorting criteria by selecting the columns (from the existing columns in the candidate data) based on a priority criteria.
    2. The user selects either ascending or descending order for each priority.
    3. The user saves the sorting preferences.
    4. User initializes the "Sort Candidate Data" functionality.
    5. System sorts the candidate data according to the user sorting criteria.
  - **Alternate Scenario**:
    1. ∗ After sorting the file, if there are multiple ties, the user is prompted to give additional rules to break the ties to proceed with the following steps.
- **Use Case** 13 *Get Offers*

  - **Primary Actor**: User.
  - **Precondition**: Use has sorted the candidate data.
  - **Main scenario**:
    1. User initializes the "Get Offers" functionality to download the offers made to the students in the particular round.
    2. The downloaded csv file will be in a format required by the CoAP portal.
  - **Alternate Scenario**:
    1. None

3.1.6 Others

- **Use Case** 14 *Update seat matrix in subsequent rounds (except Round 1)*
  - **Primary Actor**: User.
  - **Precondition**: Seat matrix has changed in between the CoAP rounds.
  - **Main scenario**:
    1. The user changes the seat matrix in between the CoAP rounds.
  - **Alternate Scenario**:
    1. * The user cannot decrease the seats in the seat matrix between the CoAP rounds. An attempt to do so will generate a prompt warning the user.

3.2 Performance Requirements

1. Should be able to run reasonably well even on a low-end laptop.
2. Should be able to handle large XL files of data, of the order of 1000s of rows and approximately 100 columns.
3. Should run in reasonable time, e.g. roughly an hour for about 2000 rows and 70 columns of student data.

3.3 Design Constraints

- ***Security***:
  1. The user information (username and password) should be encrypted and stored in the server.
  2. The data that would be uploaded to SAP should be protected from being accessed by external applications that are already present in the user system.
- ***Scalability***:
  1. SAP should be able to handle large .csv files without crashing.
  2. The SAP server should be able to serve a large number of users simultaneously ($\approx$ 1000 users).
- ***Fault tolerance***:
  1. Data should not become corrupted in case of a system crash or a power failure.
- ***Integrity***:
  1. The input data should be clean.

3.4 External Interface Requirements

1. The application opens with a login window.
2. Once the user has logged in securely, a window is opened with a sidebar that has the option to create new projects or manage existing ones.
3. In each project, 10 rounds can be accessed. A particular round can only be accessed once the previous round is done.
4. In Round 1, two csv files must be uploaded to get that round offers, namely the seat matrix csv and candidate data csv. Once the candidate data csv is uploaded, there is a window to preview the csv data. A sort button is also present, which sorts the data based on entered priorities.
5. Once the sort button is clicked, there is a pop-up that is generated that has the option to add priorities for sorting, for example which column has what priority and even the order to sort in (such as ascending or descending). Then these priorities are saved.
6. The sorted candidate data can be previewed in the window again. Once this is done, we can download the offers in the format required by CoAP.
7. In the subsequent rounds, the updated seat matrix and CoAP response .csv files should be uploaded to download that round's offer.

Chapter 4

Future Extensions

1. Tool should be able to show various statistics about how many offers are in the accept-and-freeze mode/reject-and-wait, etc., which will give the department/institute an idea of the trend across various iterations of the CoAP process.
2. Enable the tool to be able to directly send the processed shortlists directly to CoAP.
3. Handling concurrent transactions across devices for a user.
4. Tool should support shortlisting based on custom-defined rules that the user can define using a specific protocol rather than just those based on the CoAP shortlisting rule.
    - Suggest any inconsistencies that might be present in the custom-defined rules.
    - A further extension would be to enable the user to define the rules in plain English and use NLP to infer the rules being requested for.
5. Enable multilingual data to be input in the data sheets.
6. Integrate accessibility features into the tool, such as a voice assistant.
7. Automatic validation of category certificates using state-of-the-art image captioning techniques.

Chapter 5

Appendix

5.1 Appendix A: Sorting procedure used by the portal

The sorting procedure used by the portal is flexible and allows the sorting method to be declared by the user, i.e. it allows the user to decide which columns (criteria) should be used to sort the students as well as what ranking to follow with respect to the criteria (selected columns). This allows the user to modify the sorting procedure before the CoAP process starts.

5.2 Appendix B: Seat Matrix declaration and modification

The seat matrix initial declaration involves either uploading a .csv containing the seat matrix statistics, i.e. the specialization, number of seats, seat type etc., or entering the seat matrix through our portal's input interface. The seat matrix modification involves changing the statistics of the initially declared statistics i.e. increasing the seats of a specialization. We don't allow for a decrease in the seat count during the modification procedure. These two procedures together give the user flexibility, which involves accommodating for introduction of new specialization during any given year as well as accommodating seat count changes due to any other action.

5.3 Appendix C: User Interface

The description here comes in a sequel to the External Interface Requirements (3.4) described above.

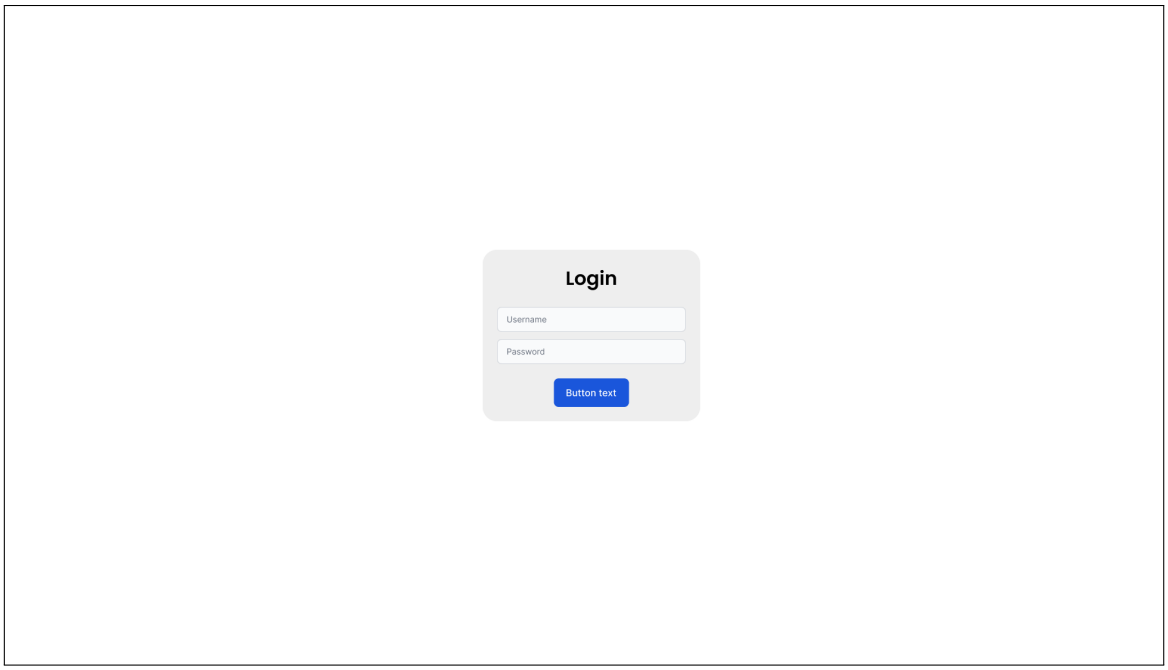
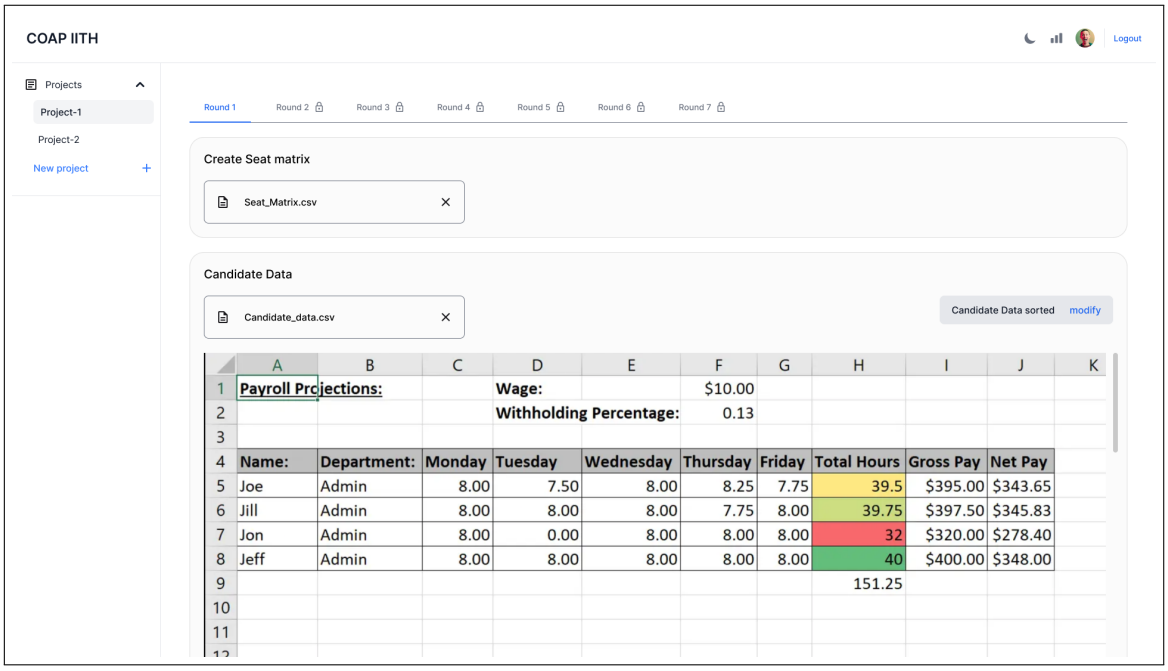


Figure 5.1: The application opens with a login window.



	A	B	C	D	E	F	G	H	I	J	K
1	Payroll Projections:			Wage:		$10.00					
2				Withholding Percentage:		0.13					
3											
4	Name:	Department:	Monday	Tuesday	Wednesday	Thursday	Friday	Total Hours	Gross Pay	Net Pay	
5	Joe	Admin	8.00	7.50	8.00	8.25	7.75	39.5	$395.00	$343.65	
6	Jill	Admin	8.00	8.00	8.00	7.75	8.00	39.75	$397.50	$345.83	
7	Jon	Admin	8.00	0.00	8.00	8.00	8.00	32	$320.00	$278.40	
8	Jeff	Admin	8.00	8.00	8.00	8.00	8.00	40	$400.00	$348.00	
9								151.25			
10											
11											

Figure 5.2: next phase

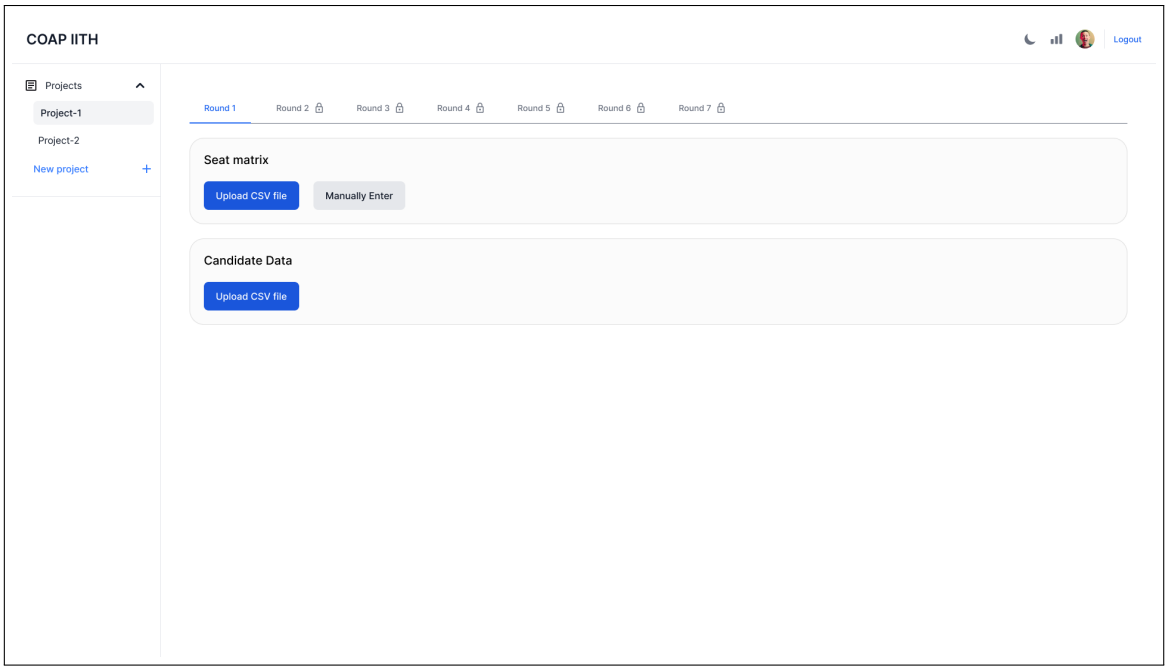


Figure 5.3: Round 1- Default screen.

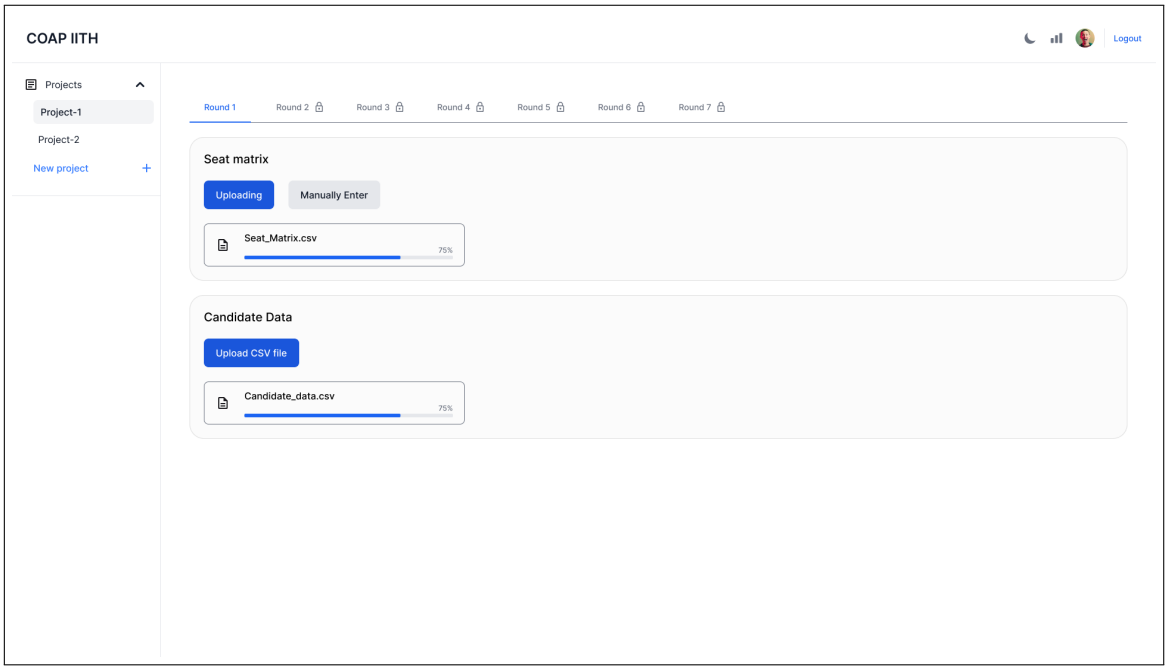


Figure 5.4: Round 1- File Uploading screen.

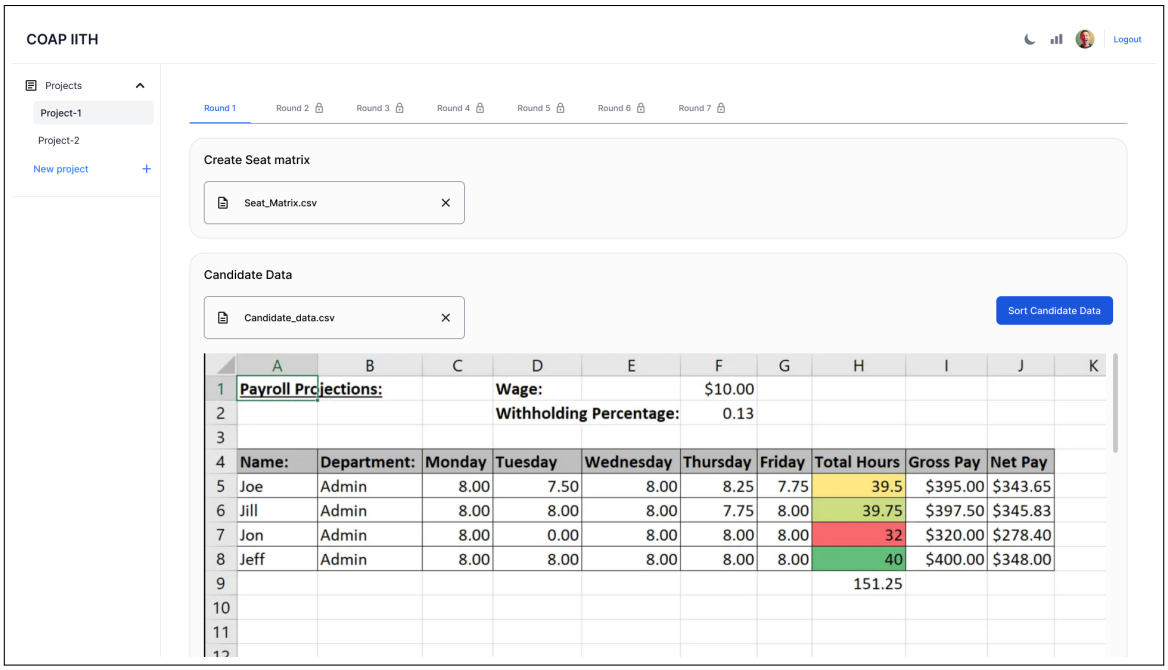


Figure 5.5: Round 1- File Uploaded screen.

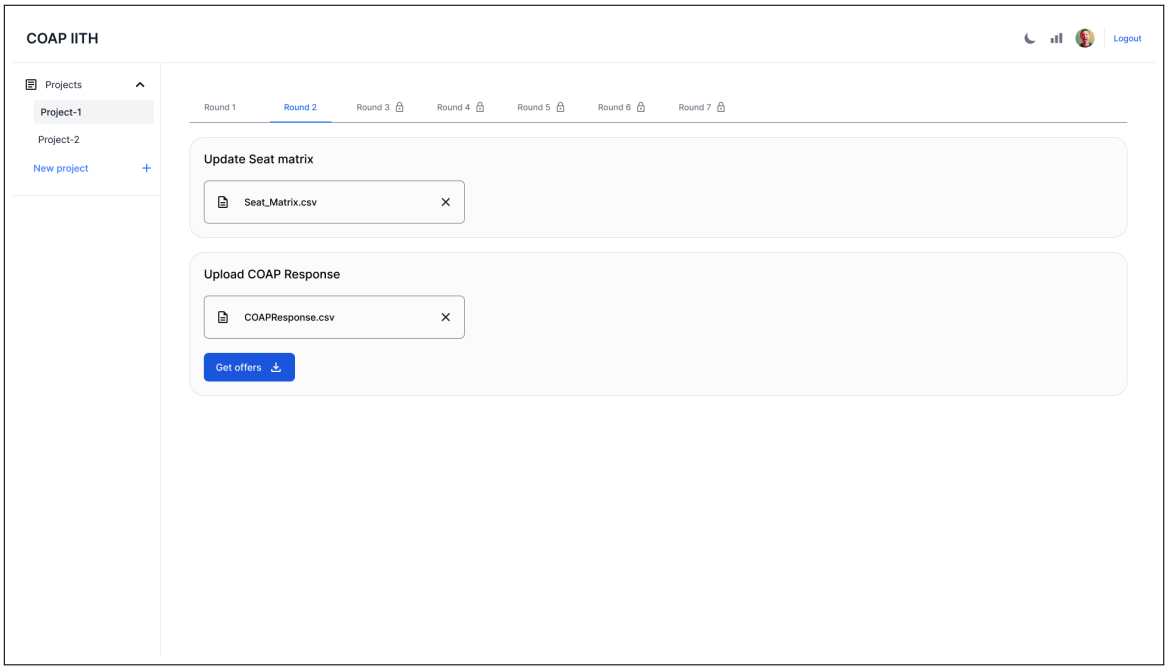


Figure 5.6: Round 2- File Uploaded screen.

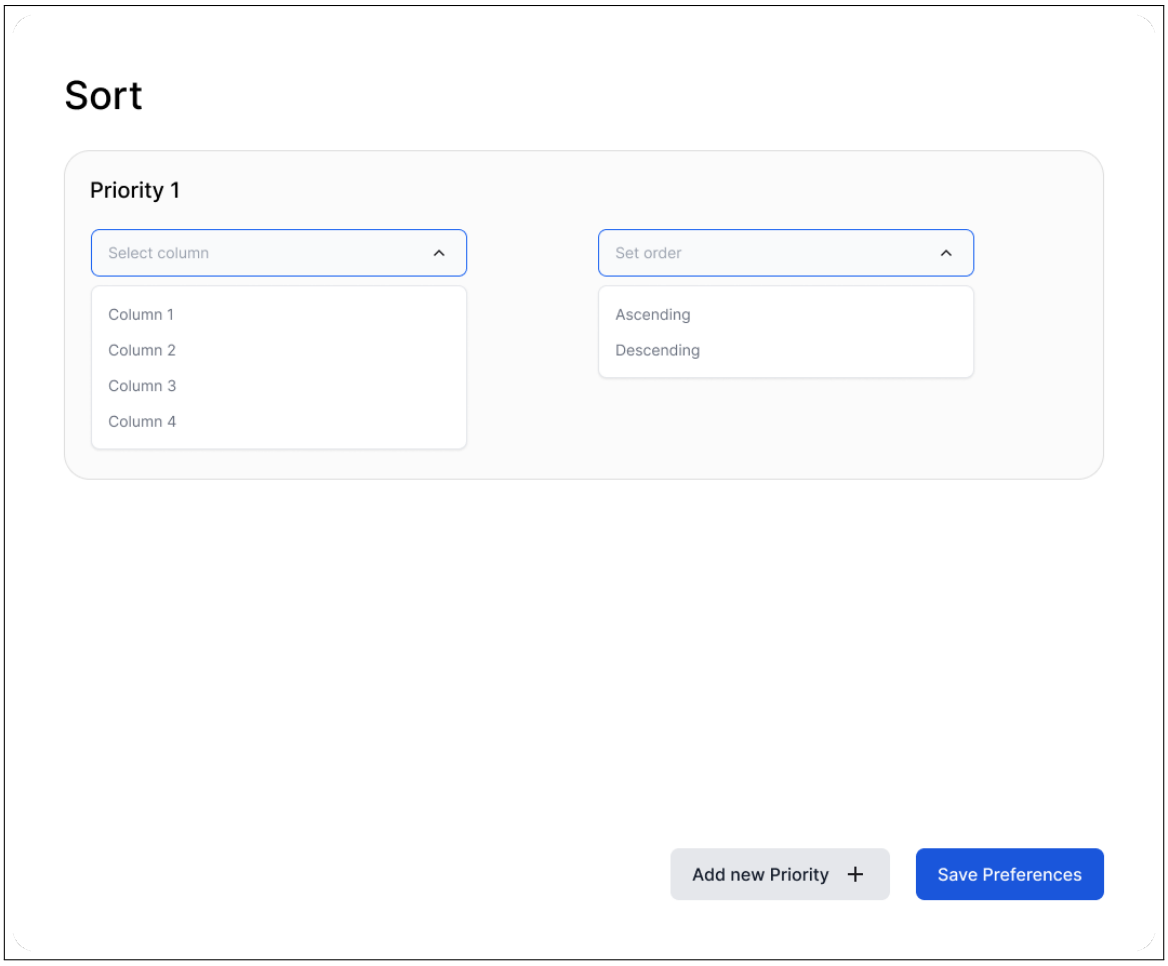


Figure 5.7: Drop Down for preferences of sorting function.

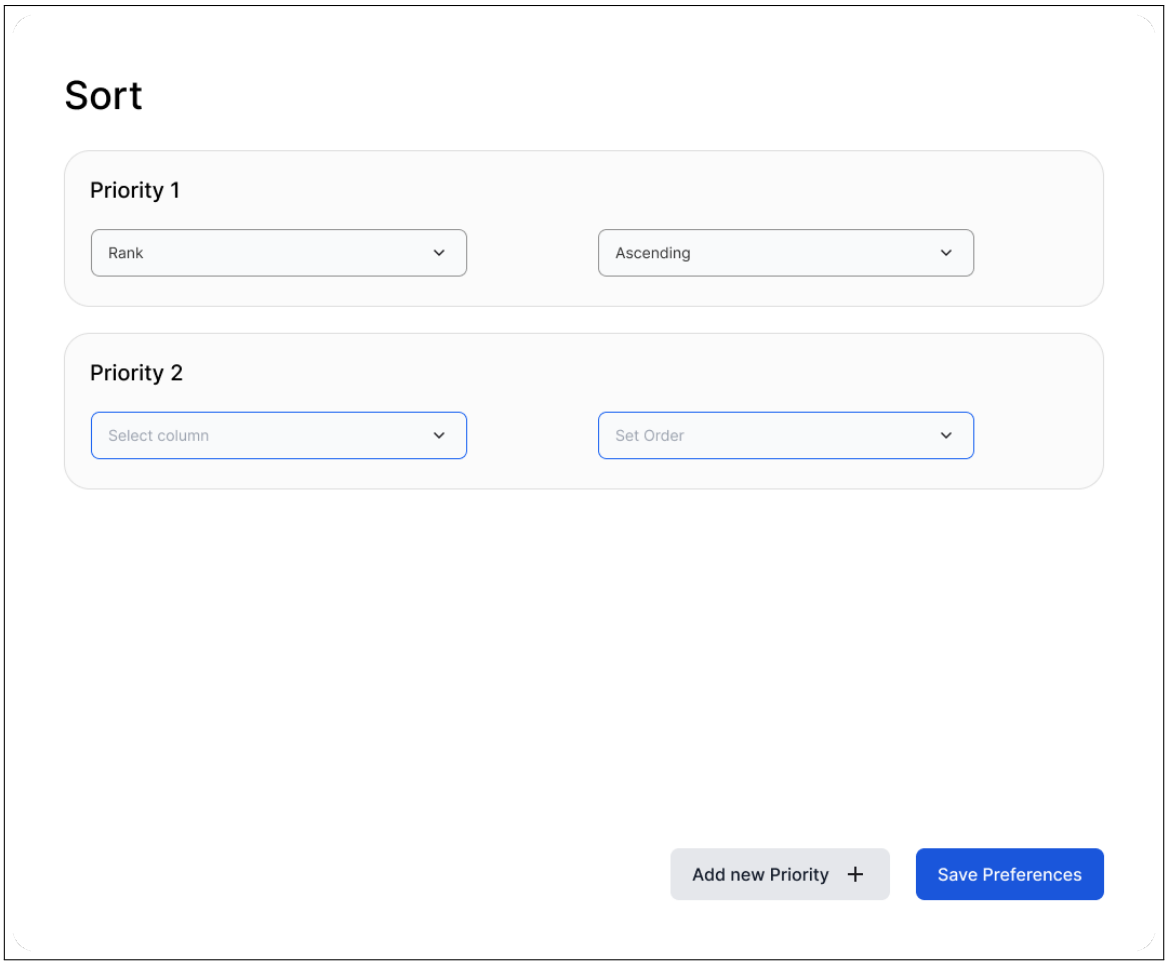


Figure 5.8: New preference added to priority screen for sorting.