*Problem Specification: Student Grade Monitoring*

**Problem Domain**

* During exam boards, advisors of studies and school administrators have records of student’s marks that need to be inputted into an internal system that gathers their marks. An advisor of studies needs to evaluate these grades and take appropriate action depending on the result of the student’s grades.
* For each module taken by the student they will be assigned a grade that is then classified which has been determined by rules and regulations of the university. If a student does not submit coursework or attend and examination, an abbreviation of that will be used for their grade.
* An advisor of studies will either assign a classification for the student’s marks based on their total grades or arrange an intervention with the student to discuss their progress. At first this will be through a letter, and then a meeting with the student, an additional Stage Coordinator, and a school administrator will be arranged.
* The school administrator will take minutes from the meeting with the student, and these will be sent to all parties.
* There is currently no centralised area where all of this information is contained and both school administrators and Stage Coordinators need to gather this information from multiple places.

**How the problem will be addressed**

* A Stage Coordinator will have a spreadsheet containing the marks of all students on their modules. The proposed system will allow stage coordinators to upload the spreadsheet and evaluate the results.
* A Webapp will be used to upload the spreadsheet into a database and extract the data. This data will then be displayed on the webapp, showing the student’s marks they have received for each module.
* The student display will contain information about the student, and the actions that should be enacted following University guidelines.
* Meetings can be created and scheduled in for students that require this, meetings will be assigned relevant staff and there will be a platform for recording meeting minutes.
* A historic view of students detailing any meeting/letters sent. There will also be minute meetings viewable for students.
* A display page will also illustrate all student’s grades with a breakdown of how many are on course/going to graduate and the number of students requiring attention.

**Proposed System Features for Different types of Users**

*School Administrator/Clerical User*

* Ability to schedule in a meeting for a student and assign a stage coordinator and a relevant academic to the meeting.
* Can view, create and update meeting minutes for students.
* Can create, edit, and delete letter templates that are sent to students. Can also then send letters out to relevant students.

*Stage Coordinator*

* Can upload csv file of student grades. Then will have the ability to update any grades that are either missing or incorrect.
* Can view a student’s grades, the actions required and whether they will be on course to graduate. They will also see whether letters have been sent out and if they have attended meetings. They will have access to the meeting minutes as well.
* Will be presented with a dashboard of their current students and given a breakdown of their performance.
* Have an event calendar that shows their upcoming meetings with relevant students and staff.

*Website Administrator*

* Will have the ability to create, update or delete staff members in the system and the ability to grant them relevant roles within the webapp.
* Can send out passwords for new users.

**Proposed Interface Elements**

* A webapp to be used as a desktop application
* There will be a main landing page that will ask a user to log in.
* Once logged in a user will have different menus on the left had side. The main page for the school administrator and the academic will be different.
* The academic’s main page will have a graphical representation of their students and a breakdown of their performance. They will also be presented with a calendar of upcoming meetings that they will need to attend.
* The school administrators main page will contain a calendar of upcoming meetings, and a Todo which will include a list of students that have attended meetings, but do not have meeting minutes attached.
* Stage Coordinators will have a tab for bulk upload of results. This will include a form to first choose which semester these results are for and the academic year.
* Stage Coordinators will have a tab for students that allows them to search for individual students and show their grade information.
* Stage coordinators will have a tab for modules that groups students within that module. These can be filterable with academic year. It will show all students enrolled in that module.
* Site admin users will have a tab that shows users and allows them to assign relevant roles and send password reset links.

**Technology Investigation**

**Front-End Frameworks**

*Benefits of using ReactJS:*

* React has access to multiple libraries that can be added to aid in development.
* React’s states allows for a responsive and quick user experience.
* Using JWT for token authentication will improve security and allow for different users to log in.
* As each component is self-contained, it leads to scalability due to only make changes in one or two places.
* When data is updated on the site, it will be rendered quickly due to React’s state rendering.
* React can be easily tested with multiple different testing libraries, this lends itself to test-driven development.
* Can utilise Chart.js for dashboard component.

*Benefits of using Tailwind:*

* Open-source library that is free.
* It is lightweight as css files are not needed when adding into react.
* Large amount of highly customisable features.
* Large documentation with detailed explanation. There is a large learning curve, but this pays off in the long term.

**Backend Frameworks**

*Benefits of using Node with Express and MySQL*

* Node has multiple libraries that will aid with csv uploads and inserting into the database.
  + Streams can be utilized for large csv files.
  + Mutler can be used for initial uploading of the csv files.
  + Fast-csv for processing.
  + Sequilize for inserting into database
  + Kue can be used to help with batching jobs in case csv processing is still being problematic.
* Can create unit tests using Jest or Supertest and can be implement for test-driven development.
* Api creation using Restful principles can be implemented and tied in with frontend.
* Mysql database used for relational database. It will allow for larges amounts of data and queriers can be linked to get relevant information quickly.

**Gnatt Chart**

**A graph with a purple line

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated  
  
A screenshot of a computer

Description automatically generated**