IT314: Software Engineering
Lab Session IV - Specifying Tools and Technology
Student Information System G23

(a) You must finalize/identify the tools, technologies, and frameworks you will use to develop/implement your project.

Technologies and Frameworks:

1. **Front-end development -** JavaScript, CSS and Bootstrap (CSS framework)
2. **Back-end development -** JavaScript, Node JS with Express JS (a web framework for Node JS) and EJS (templating language to generate HTML markup with plain JavaScript)
3. **User authentication -** Passport JS - authentication middleware for Node. js applications. With it, authentication can be easily integrated into any Node- and Express-based app.

(b) For your project, you have to use the NoSQL databases of your choice strictly, and you can also explore and use ElasticSearch (DB) Database for the same.

Database:
MongoDB and mongoose -

Reason -

- MongoDB is a popular NoSQL database that is designed to be flexible, scalable, and high-performing. It is a good choice for applications that need to handle large amounts of data and require fast read and write operations. Its schema-less data structure allows for evolution over time without needing to update the database schema. Additionally, MongoDB has a powerful query language and supports a wide range of data types and indexing options.
- Mongoose is an Object Data Modeling (ODM) library for MongoDB and Node.js. It manages relationships between data, provides schema validation, and is used to translate between objects in code and the representation of those objects in MongoDB.

Collections -

- Student login details
- Faculty login details
- Admin login details
- Student demographic details
- Academic records
- Financial/fees payment information
- Attendance record
- Student conduct record
- Course program

(c) Estimate the effort of your project and narrow down the scope based on the estimation. You can apply Function Point (for user stories) or Use Case Size Point (for use cases) for estimation.

Function Point (for user stories)

External input:

- Student registration
- Professor information
- Course registration
- Grade input
- Attendance input
- Student record update
- Input of any complaint
- Generation of grade card
- Generation of fees receipt
- Login details
- Add semester
- Add program (degree and branch)
- Assign professor to course

External output:

- Student Profile
- Exam Results
- Financial information//Fees payment
- Course catalogs

- Enrollment Reports
- Alerts and notifications
- Attendance details

Internal Files/database :

- Student login details
- Faculty login details
- Admin login details
- Student demographic details
- Student Admission details
- Academic records
- Financial/fees payment information
- Attendance record
- Student conduct record
- Course program
- Complain/feedback Record

	Measurement parameters	Example	Average Weight		Count	Weighing factor
1.	Number of external inputs (EI)	Login details, student registration, course registration, student record update, grade input, complaint input, generation of grade card etc.	4	*	13	52
2.	Number of external outputs (EO)	Student Profile, Exam Results, Financial Information, Course catalogs, Enrollment Reports, Alerts and Notification	5	*	7	35

3.	Number of external inquiries (EQ)	Here any outside institute is not involved so the number of external inquiry is 0.	4	*	0	0
4.	Number of internal files (ILF)	Database of role wise actor login details, Students personal, academic, conduct, financial, attendance details Course program details	10	*	11	110
5.	Number of external interfaces (EIF)	Here we are not using any 3rd party database/ routines/ interface.	7	*	0	0
Count Total						197

➢ **Calculation of $\sum(f_i)$:**

Here is the rating of all the 14 questionnaires on the scale of 0 to 5.

Sr. No.	Questions	Rating
1	Does the system require reliable backup and recovery ?	5
2	Are data communications required?	3
3	Are there distributed processing functions?	1
4	Is performance critical?	5
5	Will the system run in an existing, heavily utilized operational environment?	2
6	Does the system require on-line data entry?	5

7	Does the on-line data entry require the input transaction to be built over multiple screens or operations?	4
8	Are the master files updated on-line?	4
9	Are the inputs, outputs, files or inquiries complex?	3
10	Is the internal processing complex?	3
11	Is the code to be designed reusable?	5
12	Are conversion and installation included in the design?	3
13	Is the system designed for multiple installations in different organizations?	0
14	Is the application designed to facilitate change and ease of use by the user?	4
	$\sum(f_i)$	**42**

FP = Count-total * [0.65 + 0.01 * $\sum(f_i)$]

= 197*[0.65+0.01*42]

FP = 210.79

Reference:

https://www.javatpoint.com/software-engineering-functional-point-fp-analysis

https://www.codingninjas.com/codestudio/library/functional-point-fp-analysis-in-software-engineering