Campus connect

Hackathon VC competition

Team 404

Contents

Introduction 2
Our goal for this hackathon: 2
Members + Roles: 2
Functional Requirements: 2
Non-functional Requirements: 3
ERD: 4
    Link for better view: 4
Considerations: 4

Introduction

For this project, we have been tasked with creating a student hub with students can collaborate with each other. They will also be able to share materials, join groups/communities, join meetings/events and post questions, ideas or tips.

Our goal for this hackathon:

Our goal is to see how quickly we can create a large project with difficult architectures such as MVC with entity framework, dependency injection and hosting. We would also like to use this for our CV :D

Members + Roles:

Kabelo Ntokozo Will Mndebele: Main UI developer, assisting backend developer and in charge of hosting the application

Joshua Ponquett – Main backend developer, documentation specialist and GitHub setup.

Functional Requirements:

Login/sign-up: A user must be able to create an account and log into their account after its been created.

Profile creation: Users should be able to customize their profile by adding information related to their degree, skills, sports, profile photo and an other section for any left out sections.

Posting questions/ideas/tips: A forum where students can post questions/ideas/tips and get comments on their post.

Joining groups or communities: Groups/communities can be created by students but will need to be approved by an admin before members can join the group/community. These groups/communities will have an announcement section where the group/community can make announcements. They will also have their own unique message page where students can chat or ask questions.

Sharing study Material: A section where students/lectuers or admins can share study materials. This can be implemented through groups/communities or through a share study material page. A verification system to make sure these materials are safe would be nice. However some students don't want to wait 3 to 5 days just for their document to be uploaded. For this reason a report button will be implemented instead.

Report study Material: As mentioned above in the sharing study materials function, a button will be implemented which will allow students/lecturers to flag an inappropriate document. Admins will then check if these documents are safe or not and delete the post if they are deemed unsafe.

Scheduling meetings: This will be done through events which students can join. If they join it they are essentially RSVPing for the event. They will no longer be able to join the event/meeting after a certain date. They will be able to remove their RSVP at any time as it is not easy to predict what the future will hold and sometimes something just pops up.

Non-functional Requirements:

Availability: Website must be hosted online and available 24/7

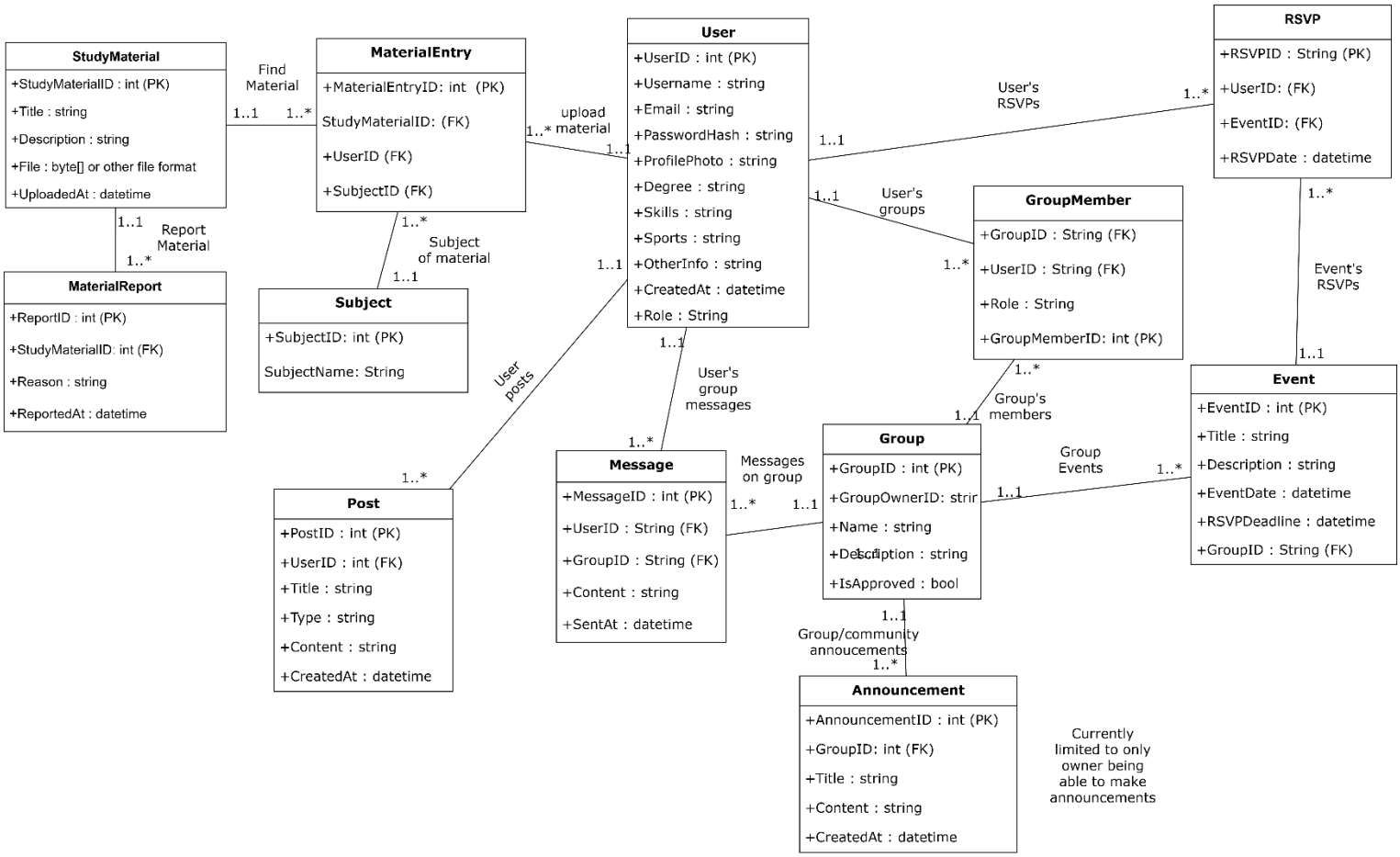
Compatibility: Website must load correctly on mobile and computer devices.

Maintainability: The website will be written with MVC (model-view-controller), Entity Framework And Dependency Injection (repository + interface pattern) so that it can be easily updated and maintained.

Scalability: Limited scalability since we are using built-in sqlite database for ease of hosting. This can easily be updated by changing the program.cs file. The addDbContext in program file would need to use a connection string to connect to an sql/nosql cloud server instead to make this project more scalable.

Security: passwords are hashed before being stored in the database and unhashed after being stored. Authentication and authorization systems have been put into place to restrict visitors from posting and to restrict users from accessing admin pages.

ERD:



Link for better view:

https://drive.google.com/file/d/1WInvtsE2RU9FikPB8F0FtBKo3DV_mGH2/view

Considerations:

SQlite limitations: SQlite allows for many reads at the same time but doesn't allow for many inserts at the same time. So essentially a queue system would need to be created for inserts. This would be a problem if the actual student hub went live. The database is fairly easy to change but it would still require a lot of consideration.

Group/community announcements: Its currently restricted to where only the owner of the group can make announcements in the group/community. If this were to change you would need to add a joining table between the two to prevent a many to many relationships.

Comments not implemented for posts: Current ERD does not consider comments from users on posts created by the user or other users