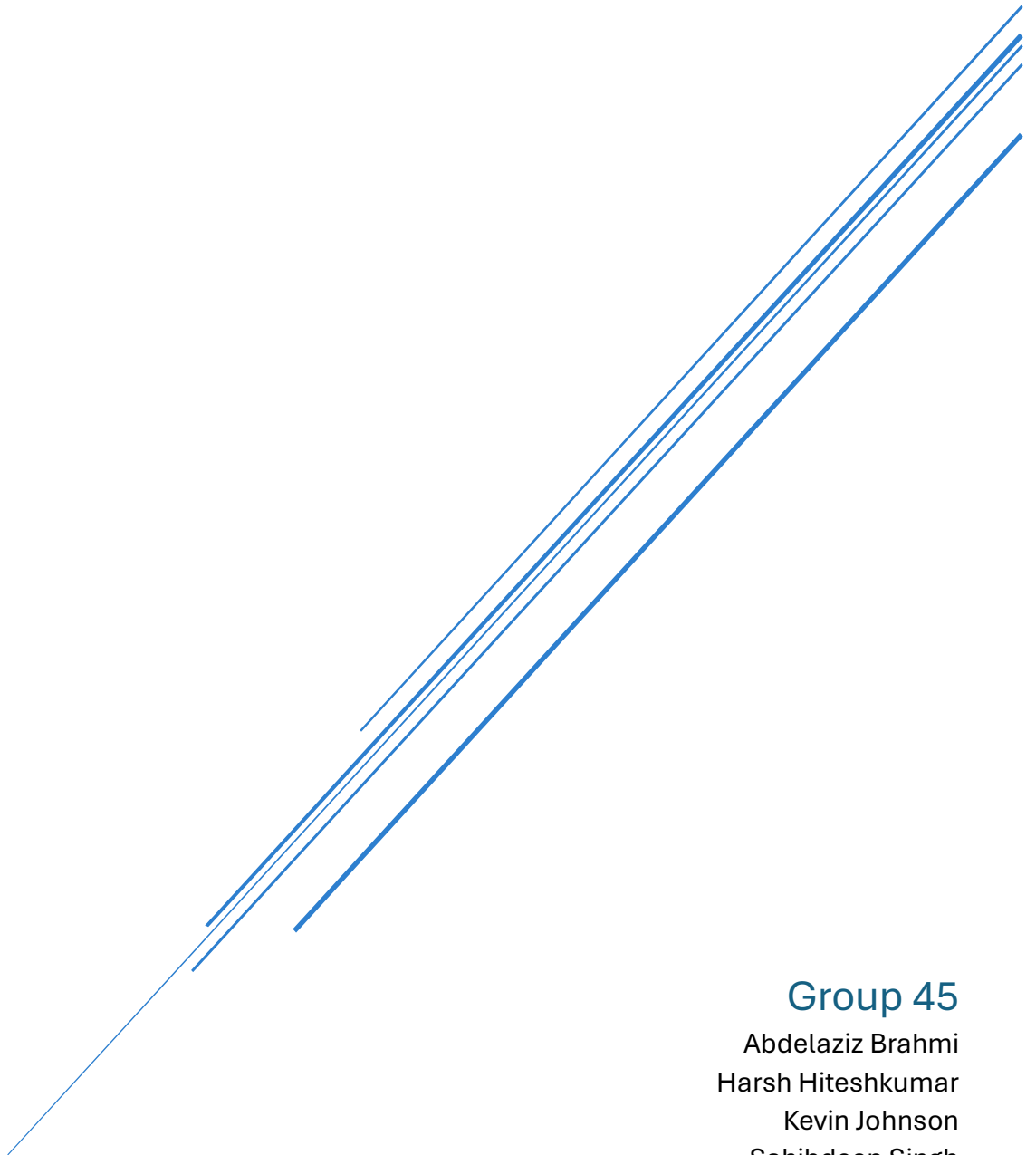


Coursework 3

File Repository System



Group 45

Abdelaziz Brahmi
Harsh Hiteshkumar
Kevin Johnson
Sahibdeep Singh
Winston Ung
Matthew Williamson

Coursework 3
File Repository System

Introduction	3
Stakeholders	3
User	3
Admin	3
Group Member	3
Group Leader	3
Investors	3
Software Developers	3
Support Team	3
Concerns	4
Architecture Views	5
Decomposition View	5
Diagram	5
Rationale of the diagram	5
Dependency View	7
Diagram	7
Rationale of the Diagram	8
Execution View	9
1. Create File	9
Normal Flow	9
Alternate Flow Diagram	10
2. Edit File	11
Normal Flow	11
Alternative Flow	12
3. View Metadata	13
Normal Flow	13
Alternate Flow	13
Security Review	14
Access to files	14
Uploading files	14
Sending Notifications	15
Group Management	15
Managing Shared Devices	15
Appendix	16
How we worked as a group	16
Timeline	17

Coursework 3
File Repository System

Week 1 11 / 11 / 24.....	17
Week 2 18 / 11 / 24	17
Week 3 25 / 11 / 24	17
Week 3 28 / 11 / 24	17
Contributions	18

Introduction

The File Repository System is basically a tool created for a local government department to help them manage all The Software Architecture document provides a brief and detailed description of the development architecture, which provides a comprehensive overview of the system's design. It provides interaction between such stakeholders, guarantees compliance with business and technical specifications, and acts as a starting point for future enhancements as well as maintenance of the software.

The File Repository System is basically a tool created for a local government department to help them manage all kinds of documents, like planning applications. It's built to handle different versions of files, no matter the format, and keep all the related details (metadata) in one place. Some of its main features include version control, comment box for the users, file histories, and retrieving a file. It also tracks the file owner, edit made on the files and when it was submitted

Stakeholders

User

Person with file read/write/create permissions, system interaction.

Admin

Person with system-wide administrative privileges, who can manage users. Individual with full system access and control.

Group Member

Person who can access the user's submitted files to view, edit and update the submission,

Group Leader

Individual with authority to manage users and files within a group (group members) with extra privileges less than an admin

Investors

The investors are the people funding the development of this application. In the project document, they were referred to as the "local authority department", "Ms. Charlton" and "Mr. Mann".

Software Developers

The software developers are the team of people creating the application.

Support Team

The support team are people who answer questions a user of the application might have or troubleshoot problems that a user is experiencing with the program.

Concerns

Concerns	User	Admin	Group member	Group leader	Investors	Software Developers	Support team
Ability to store multiple versions of the same file so I can see a progression of change in files.	X	X	X	X	X	X	-
Ability to provide comments when a file is added so that I can give feedback on updates	X	X	X	X	X	X	-
Ability to submit files which present change so that I avoid duplication of data.	X	-	X	X	X	X	-
Being a member of groups so that I can access and submit appropriate files	X	-	X	X	X	X	-
I want to be able to access only the files stored by me or other members of my group so that I see only relevant content.	X	-	X	X	X	X	-
I want to be notified by email if a file from one of my groups has a new version, so I am kept updated when a file has been changed	X	-	X	X	X	X	-
I want to see information about the file including the file owner, who made the latest edits and the date of submission so files can be appropriately traced by relevant people	-	X	-	X	X	X	-
I want to be able to provide a single comment which is applied to multiple files so that relevant comments can be added in bulk to increase productivity	X	-	X	X	X	X	-
I want to be able to change group ownerships for multiple files so that relevant permissions can be applied in bulk to increase productivity	-	-	-	X	X	X	-
I want to be able to access reports on individuals and groups so that I can keep up to date with what users are working on	-	X	-	X	X	X	-
I want to have access to an archive of files that have not been access for a year or more so that files not regularly used do not clutter the File Database	-	X	-	-	X	X	-

Coursework 3

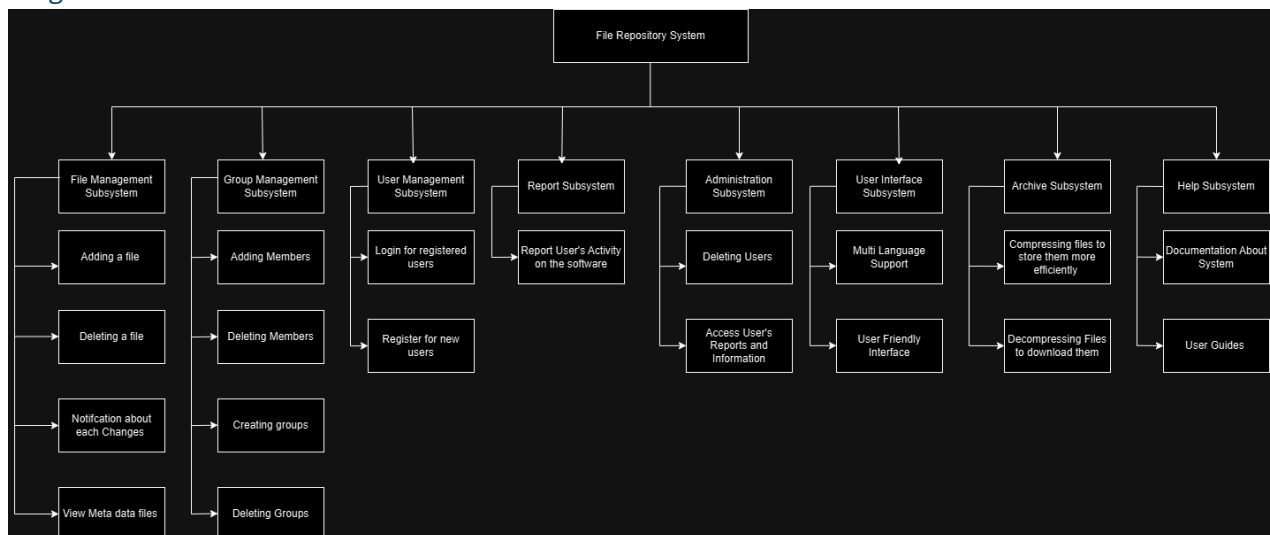
File Repository System

I want to be able to see all files and comments so that appropriate support can be provided to system users when required	-	X	-	-	X	X	-
Speed of Program	X	X	X	X	X	X	X
Easy Interface	X	X	X	X	X	X	X
Security of storage of files	X	X	X	X	X	X	X
How and What languages will it be programmed in	-	-	-	-	-	X	-
Type of application (website, mobile app....)	-	-	-	-	X	X	-
International Languages support	X	X	X	X	X	X	X
Easy and smooth customer support communication	X	X	X	X	X	X	X
Changeable Interface to someone's preferences	X	X	X	X	X	X	X

Architecture Views

Decomposition View

Diagram



Rationale of the diagram

The file repository system is broken down into multiple sub-systems. We have split the entire system into these specific sub-systems for the following reasons:

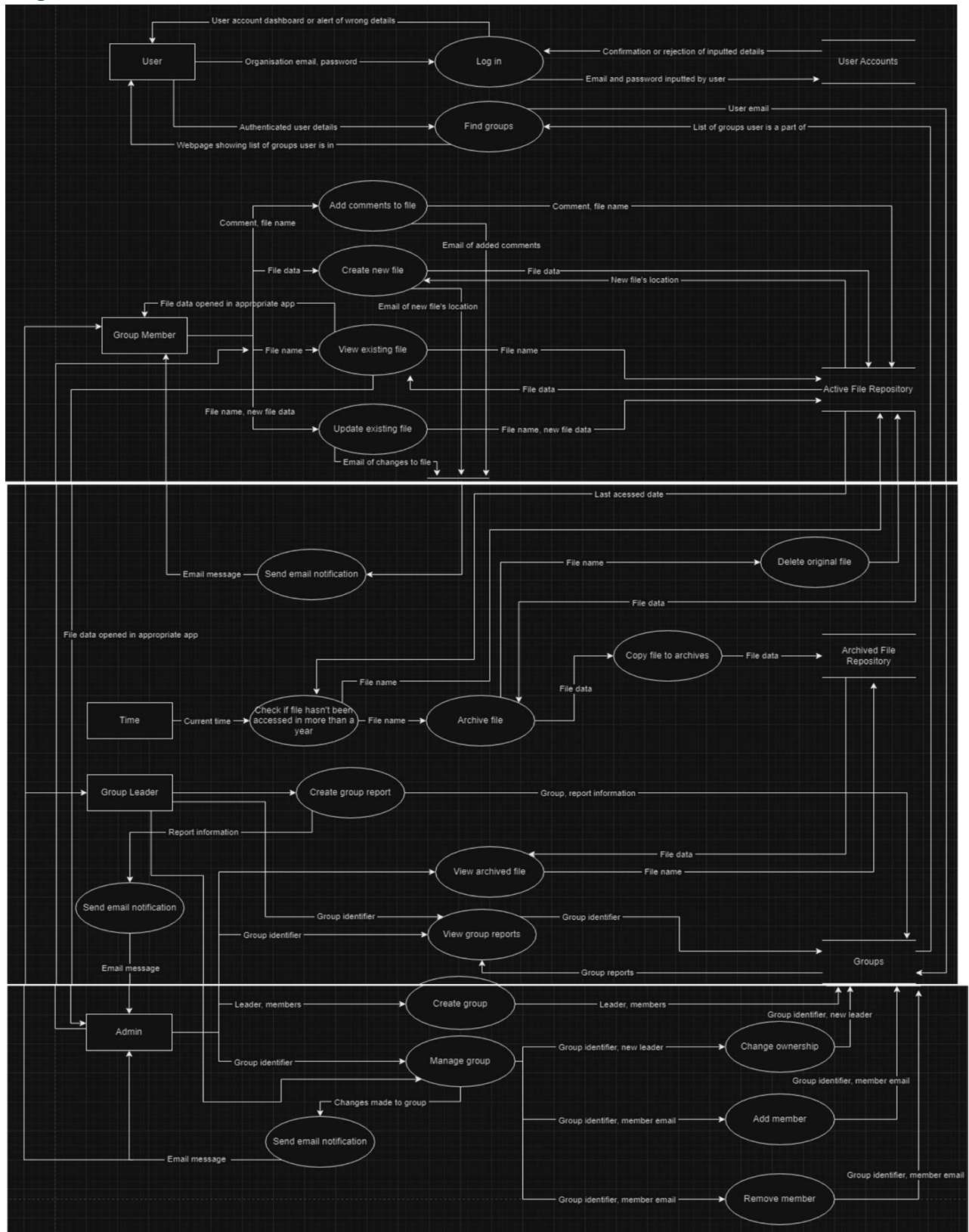
Coursework 3
File Repository System

File Management Subsystems	This is the foundation for the software. We wanted the file management sub-system to have the same features as Windows File Explorer (adding/removing files, viewing metadata), as well as a notification being sent once a file is updated or added.
Group Managements Subsystems	Users being split into one or more groups was valuable to the stakeholders of the software. We thought about the different permissions each person might have (admin, group leader, group member) and allocated different procedures that accommodates to each user.
User Management Subsystems	We thought a login system would be more practical for users than a user's data being stored on only one machine. Hence, we needed an administrator to manage inviting users to their file repository system, as well as a service for all users to create an account and login to the system.
Report Subsystems	A short summary of users' activity should be submitted to the group leaders and administrator.
Administration Subsystem	Deleting users is important when someone no longer works at the company. Additionally, administrators might want to view information and previous reports on any specific user in the company.
User Interface Subsystem	We thought the file repository system should be friendly and easy to use through the user interface. We also thought an option for multiple languages would be ideal for customers in other countries.
Archive Subsystem	The stakeholders mentioned archiving files after a year of no access. Compressing those files would be efficient on storage of the system. A decompressing tool for those files would be useful in case a file is wanting to be brought back from the archive.
Help Subsystem	If any user is having trouble with the software, a help option can be accessed for technical support and/or user guides.

File Repository System

Dependency View

Diagram



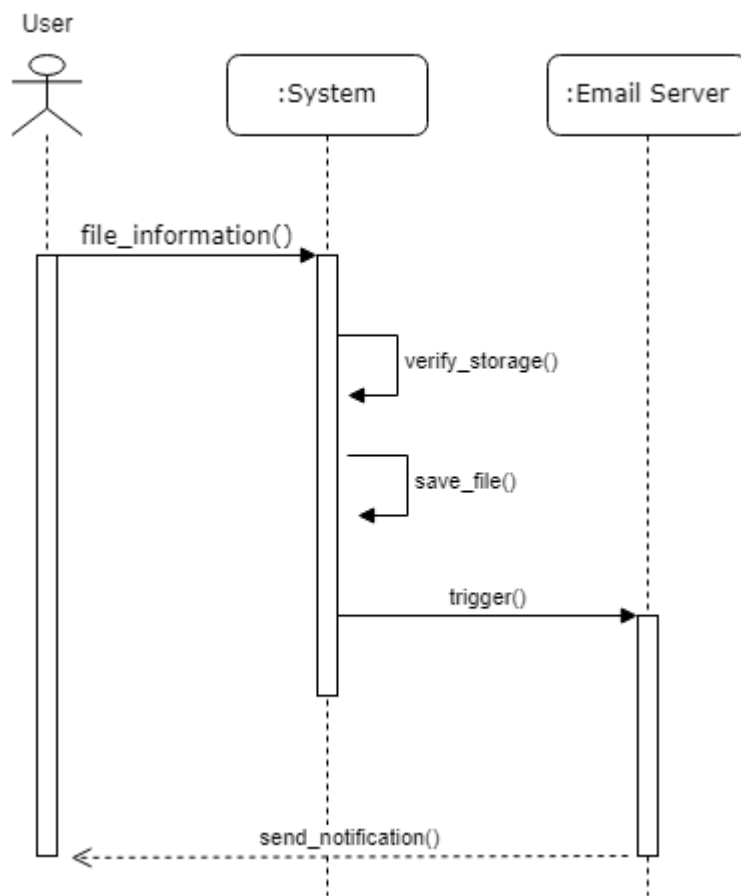
Rationale of the Diagram

The system was broken down in the dependency view diagram into many small components based on their functional and operational requirements and then designed according to the dependencies of each component. The rationale behind why the system was broken down into smaller components was to ensure that diagram provides a straightforward structure for understanding system functionality, and most importantly to see the functional unit's interdependencies. To begin with, we decomposed the file management component into smaller units such as adding comments to files, creating new files, viewing existing files and editing files in our dependency diagram because managing files is the central goal of the repository, and these different functions have different dependencies which need to be clearly shown in our dependency diagram. Therefore, this functionality is segregated into multiple units. For example, a member who wants to view an existing file will need approval from the admin so we can say that the viewing existing files unit is dependent on the admin. Contrary to this, if a member wants to create a new file, they do not need approval from the admin so they are not interdependent. Although both functional units can be accessed by group members, difference in dependencies sets them apart. We set the active file repository as a component because accessing files is the core part of the system and is dependent on many of the functional units of the file management component and for some units it is interdependent. Files not accessed for a year are automatically shifted to the archived file repository hence we created a component of the archived file repository. Like the file repository it is also dependent on the functional management component but not directly because files are first searched in the file repository and if they are not found then there is check made to see if the file has been accessed within a year from the current time. If yes, then the file is searched in the archived file repository and displayed once found. We included a notification system component known as the send email notification which has been used widely in our dependency view diagram because there is constant communication imparting between users, group leaders and admins. For instance, a report written by a group leader will be sent to the admin via email notification. Since this functional unit or component is commonly used in our dependency view diagram, its dependencies also change vastly throughout our diagram depending on who is sending the email. The admin is a significant figure in the file repository system because it has great number of duties like approving or disapproving user requests to view files, checking group reports compiled by group leaders and many more. Due to this we made the admin a component in our diagram. The admin is mainly dependent on other components who are subordinate to it in terms of hierarchy such as group members and group leaders. The group members and leaders being of considerable importance were also set as components, but they were predominantly dependent on answering or replying to emails sent by their superiors. In Conclusion, the component breakdown in the dependency view diagram made it more informative and meaningful. Some components were further broken down into units, so they are more understandable. In addition to this, the breakdown also helped in understanding how a change in one component can impact other components due to dependencies.

Execution View

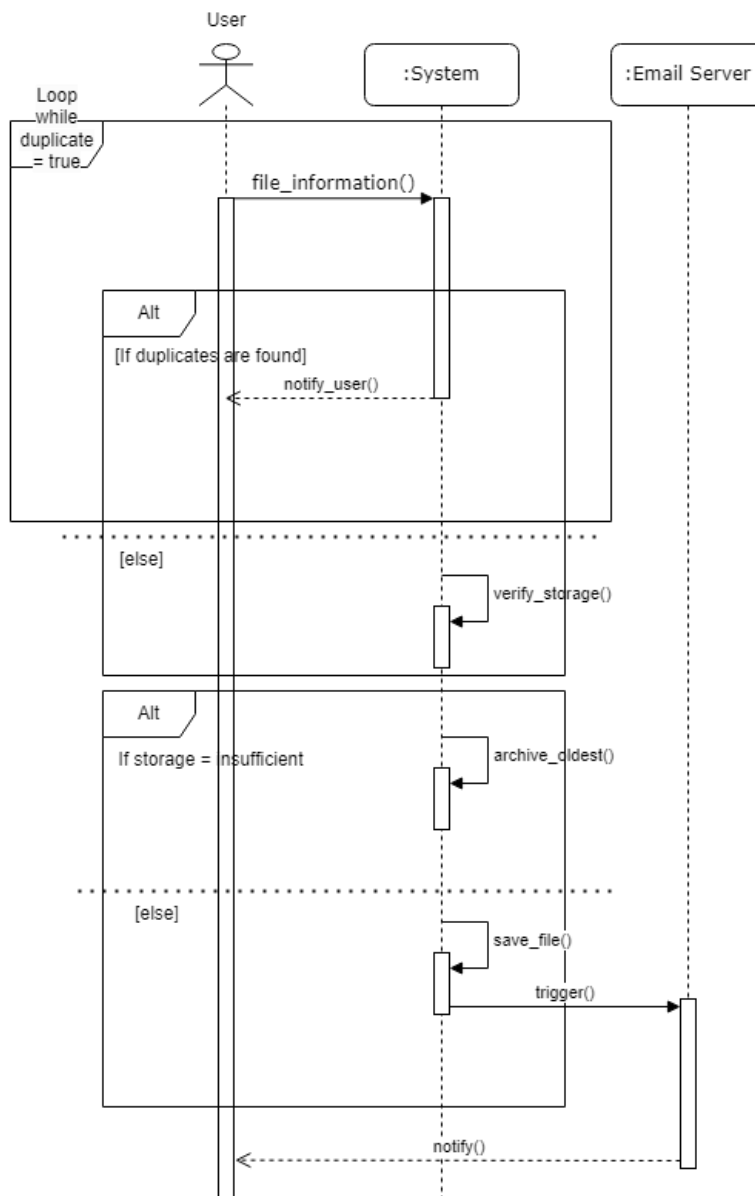
1. Create File

Normal Flow



Coursework 3
File Repository System

Alternate Flow Diagram

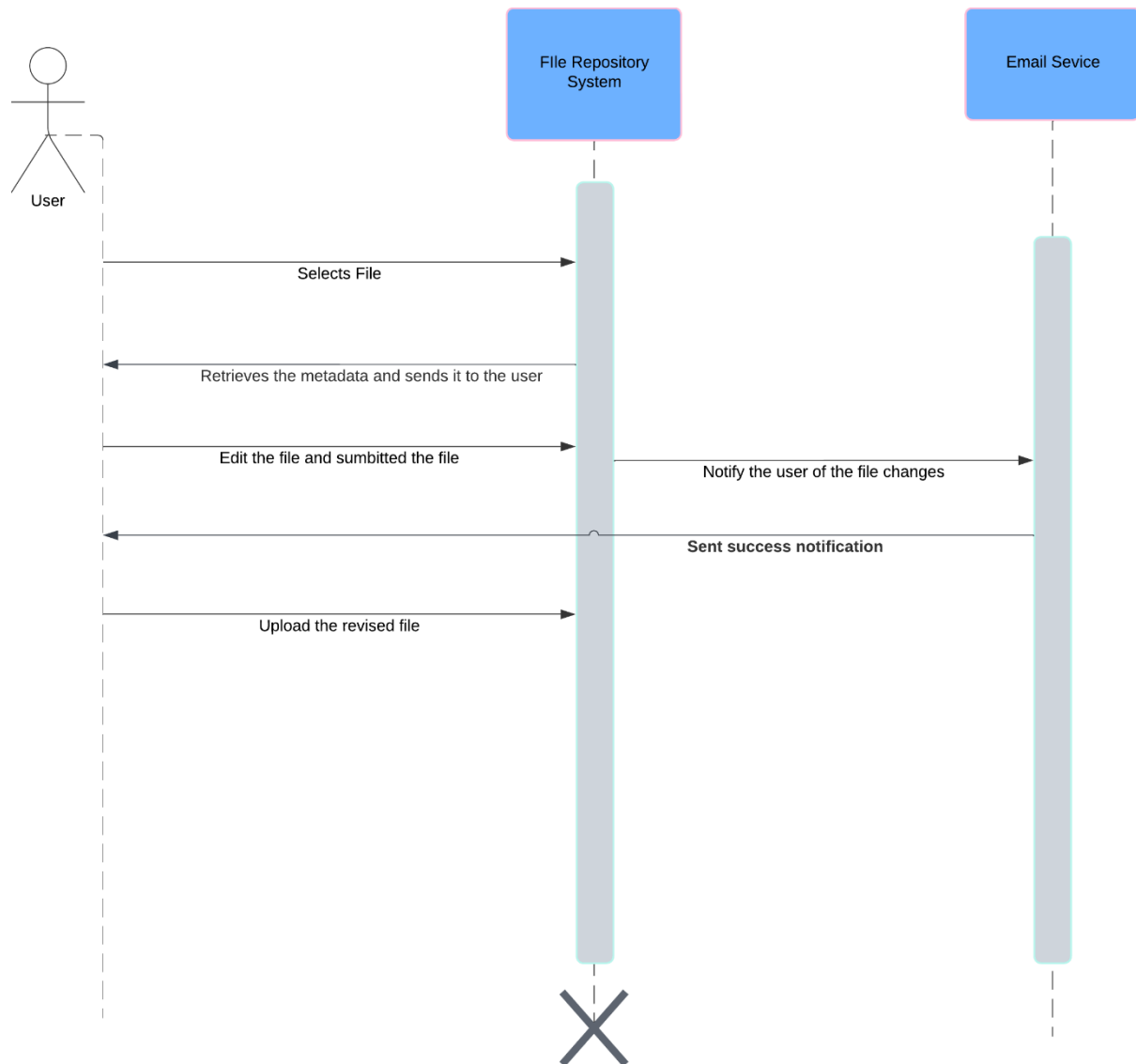


Coursework 3

File Repository System

2.Edit File

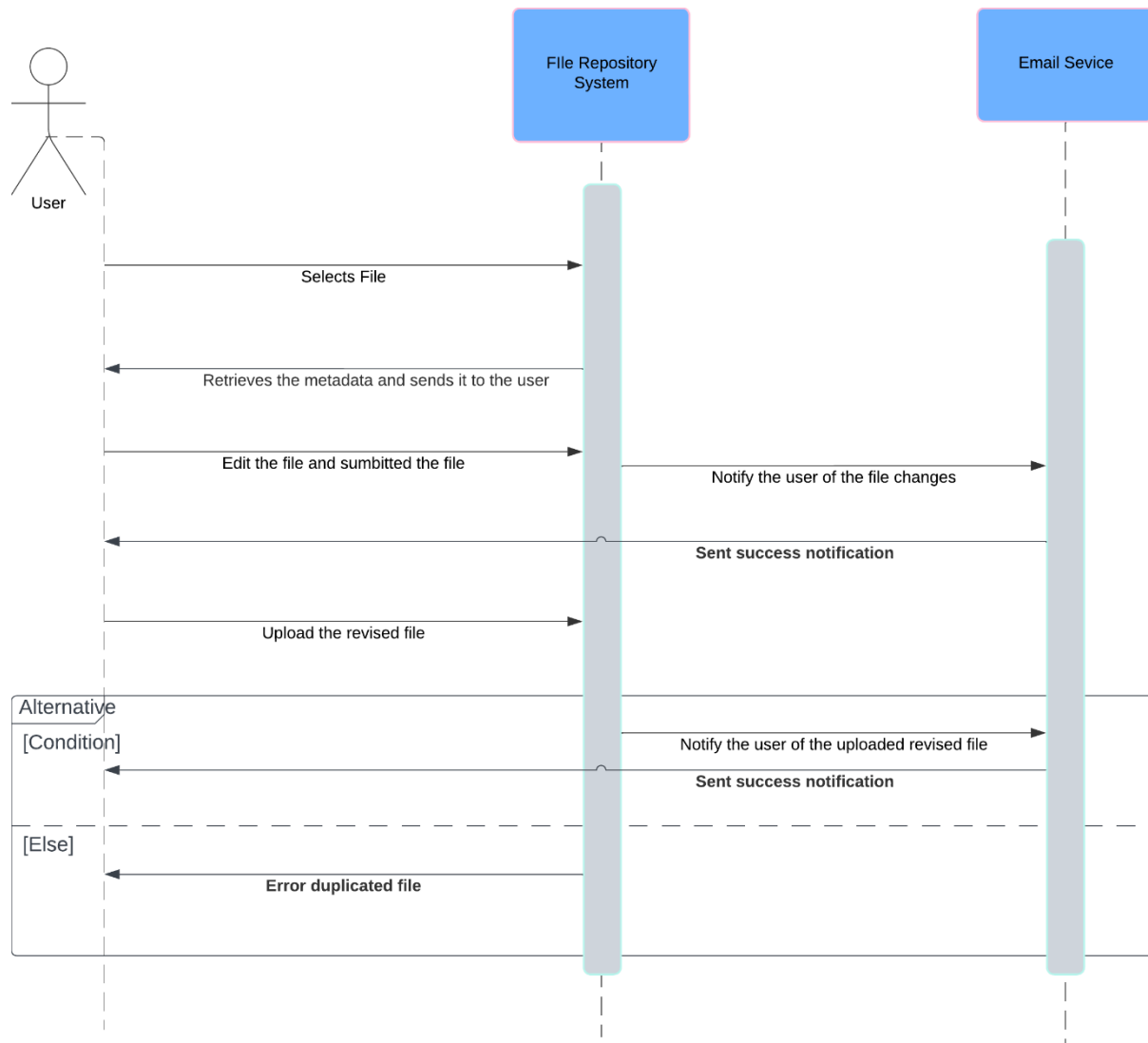
Normal Flow



Coursework 3

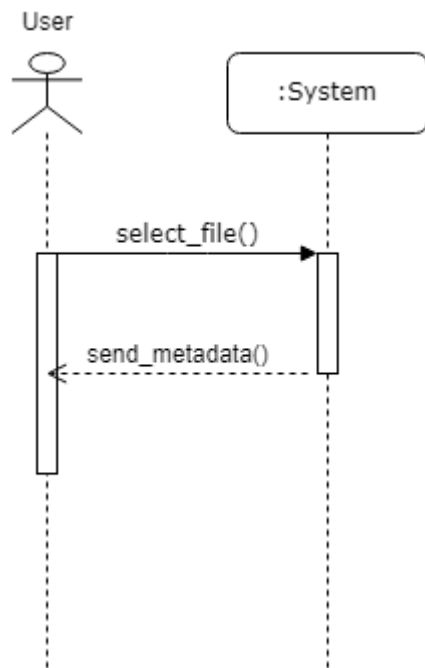
File Repository System

Alternative Flow

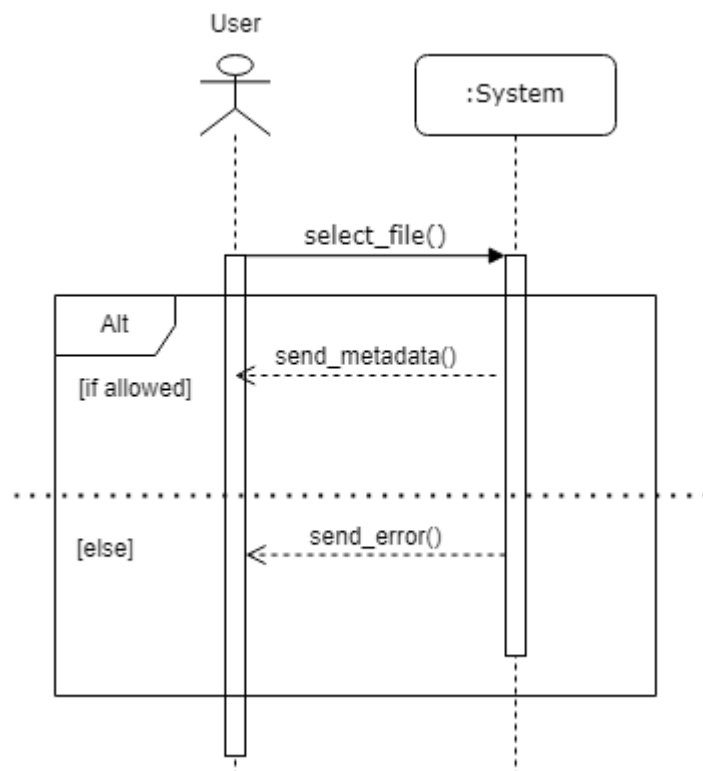


3. View Metadata

Normal Flow



Alternate Flow



Security Review

Access to files

The system requires that files uploaded to the database only be accessed by members of the group those files belong to.

To ensure that no unauthorised parties can access a group's files, a user must first be logged into their organisation account.

For higher-level actions, including updating files and comments, a user must be connected to the organisation's network.

This means files cannot be tampered with remotely, as a person must be on site to connect to the organisation's network.

The repository will require policies to restrict execution of queries to only authorised members.

For read queries, a user must simply be authenticated with their organisation's login details.

For updating files and writing comments, only queries sent from within the organisation's network shall be allowed.

This level of restriction is a compromise between ease-of-access and safety of files, as it allows users to read files within their group from any remote location, while preventing tampering of files by a malicious third party remotely, at the expense of being able to update files from any such remote location.

Uploading files

If a malicious third-party gains access to an organisation account and can upload files, these files are likely to be infected.

Users in the same group will be notified of the uploads, and may download the files on their system, which may cause the user to lose access to their files and will help spread the malware to other users.

If the malware is downloaded onto a computer connected to the organisation's network, it is possible for the malware to spread to all other computers on that network.

Consequences include losing files and exposing information of users who have uploaded files to the repository in the past.

To avoid this issue, all files uploaded to the repository shall first be scanned to ensure each file is safe.

To reduce the chance of a malicious file being missed, multiple layers of scans from different security vendors are required.

Files may flag the scanner as false positives. Rather than rejecting the upload, any files that have flagged the scanner but may not be malicious should be placed into quarantine.

Admins will be alerted to review the files. To ensure these files are not malicious, the admins should run and monitor the files in a sandbox testing environment. Once a conclusion has been made, the admins can choose to accept the files or delete them.

Any files that are confirmed to be malicious should trigger an investigation carried out by the admin into the uploader of the files.

Sending Notifications

The notification system is designed to alert or notify users about changes or updates to files of their groups. Although this feature does improve user awareness, it can increase potential security risks if the notifications are sent to incorrect users. Confidential and private data may be leaked and consequently may fall into the wrong hands.

To eliminate this risk, several steps can be taken. To begin with, ensure the notifications are only accessed by authorised users by locking the notifications so users will have to input their passwords to access the notification. The passwords will have to be strong, so they need to be made up of numbers, lower and upper case characters and other symbols. This can significantly reduce the risk of data leaks and therefore protect the system against security risks.

Along with this, sending sensitive or confidential information on notifications should be restricted or minimised. This can greatly reduce the threat posed by data leaks because if a restricted user does get a hand on the information, then they will be unable to do any further harm because the information or data will not carry any confidential or personal details.

Group Management

If there are groups made in an organisation, then verifying only authorised users such as the admins or group leaders can add, remove or change group members is extremely important. This is because if a prohibited individual manages to get access to modifying a group, then they will misuse their access by removing all the group members or by randomly adding restricted individuals. This poses a security risk as restricted users that have intentionally been added will try to gain access to private information such as passwords and usernames of other areas in the organisation from the authorised members already in the group. Since the authorised members think that these individuals are authorised, they will share this private information with them.

To prevent this from happening, we need to implement multi-factor authentication which will ban unauthorised users from modifying groups. Authorised Users such as admins or group leaders will need to keep a log for all group changes. This will include members that have been added or removed in a certain group and along with this, they will also have to provide a reason for this change and admins or group leaders who have made this change will need to write their names down as well.

In addition to this, ensuring users are only able to access files shared between their groups and not others are crucial. This is because some files are highly confidential and users higher up in the hierarchy such as Organisations leaders are the only ones that need to know of the contents of such files. Exposing such files to users that are not permitted to view these files can lead to inside threats in which the users can misuse their data. Therefore, authentication needs to be set up on all files. This means group members will only be given passwords to files that they have access to. So, users are unable to access files for which they have not been given passwords.

Managing Shared Devices

In an organisation, there are potentially serious security risks while shared devices are used by multiple users. For example, users leave their session logged in, allowing other users to access their files or any sensitive information. This could have resulted in updating records changed, gaining access to closed files and data disclosures.

To prevent these risks, the software should have a session timeout feature which logs out the user automatically after period of inactivity. Users should have their personal logging credentials as this will ensure creditability and prevent unauthorised access to the system. These safety features ensure that shared devices are secured and reduces the risk of data breaches which could compromise the organisation.

Appendix

How we worked as a group

In our group project to create a Software Architecture Document (SAD) document, we made sure that everyone had an equal opportunity to participate and worked well together as a team.

At the start, we split the sections between ourselves in which the smaller sections such as the introduction and the appendix were completed by one member each.

However, for the longer and more detailed sections such as the different type of views, we split it into groups of 2. Where each group did one of the different views.

We decided which member did what by holding an open discussion where each teammate shared their strengths and weaknesses on a specific topic.

For example, if someone were more confident with Dependency view compared to Execution View then they would be doing the Dependency View.

From the coursework before it, we still had a WhatsApp group to allow members to communicate any changes or delays in finishing their section, support anyone who was struggling with their section by sharing ideas on the chat and to give feedback on the completed sections.

This made sure everyone had a say in each section even if they were not doing that specific section.

We also set due dates for each section which ensured work was being done efficiently and allowed other members to see progress was being made.

Ultimately, this allowed us to avoid a last-minute rush to finish and guaranteed us an outstanding grade.

To sum up, we met up in our Friday Software design session to make sure everyone was satisfied with the finished SAD document before it was submitted.

Timeline

Week 1 | 11 / 11 / 24

- Creation of the document.
- Dished out tasks to each member of the group.
- All members attended the first lab session on 15/11/24.
- Individual contributions:
 - **Introduction:**
Harsh Hiteshkumar
 - **Stakeholders and Concerns:**
Abdelaziz Brahmi
Matthew Williamson
Winston Ung
 - **Security Review:**
Kevin Johnson: Access to files, Uploading files

Week 2 | 18 / 11 / 24

- **Execution View:**
 - Matthew Williamson: 1. Create File, 3. View Meta Data
- **Appendix:**
 - Matthew Williamson
- **Decomposition View:**
 - Abdelaziz Brahmi: making an approximation of how the diagram will look like (sketch)







Week 3 | 25 / 11 / 24

- **Execution View:**
Harsh Hiteshkumar: 2. Edit File

Week 3 | 28 / 11 / 24

- **Decomposition View:**
 - Winston Ung: Finishing the rationale of the diagram
 - Abdelaziz Brahmi: Finishing listing the Subsystems and the Decomposition view Diagram
- **Security Review:**
 - Sahibdeep Singh: Sending Notifications & Group Management
 - Harsh Hiteshkumar: Managing Shared Devices
- **Dependency View:**
 - Kevin Johnson: Data Flow Diagram
 - Sahibdeep Singh: Rationale of the diagram

Contributions

Member name	Contribution (%)	Signature
Abdelaziz Brahmi	100/6	
Harsh Hiteshkumar	100/6	
Kevin Johnson	100/6	
Sahibdeep Singh	100/6	
Winston Ung	100/6	
Matthew Williamson	100/6	
	100%	