Table of Contents

1 Abstract

The report presents a comprehensive exploration into the development and evaluation of an educational application which name TutorKit. It begins with a literature review that delves into the educational technology domain, highlighting the importance of various technologies and methodologies like programming languages, databases, Waterfall, and Agile. The requirement analysis section examines similar educational platforms such as Edmodo, Canvas, and Schoology to identify key features and requirements for the application. This analysis culminates in defining project, solution, business, non-requirement, and stakeholder requirements. The software design phase sketches out the system's architecture through use case diagrams, ERDs, flowcharts, and wireframes, providing a clear blueprint for development. Implementation details the tools and technologies used, including Android Studio, LDPlayer 9, and Samsung Galaxy J2 Prime, along with key code snippets showcasing essential features. The evaluation section discusses the project's outcomes, testing strategies, and results. The report concludes by reflecting on the project's achievements and challenges, underscoring the potential of the developed educational application to enrich learning experiences. The references section lists the sources and tools referenced throughout the report for further exploration.

2 Introduction

TutorKit is an innovative educational platform designed to connect tutors and students in a seamless and efficient manner. Catering to two primary roles – Tutors and Students – the application offers a range of features tailored to their specific needs. Tutors can manage their teaching schedules, track student progress, and create tests to assess understanding. Students, on the other hand, can choose their preferred tutor, access class schedules, view grades, submit assignments, and pay tuition fees, all through a user-friendly dashboard. With TutorKit, we aim to simplify the tutoring process, foster effective communication, and create a productive learning environment for both tutors and students, enhancing the overall educational experience.

3 Literature Review

3.1 Domain

In the ever-evolving world of educational technology, there are always new things to explore and discover. A Student Management System (SMS) is software that coordinates and organizes learner data of all ages through a centralized location accessible to all departments. This system allows tutors to address student requests. The student management system is designed to make tutors' lives better. It brings great value to the teacher's management and learning process and even helps increase the number of students per tutor. According to the report, the global SMS systems market size is valued at USD 7.41 billion in 2021 and is estimated to grow at a CAGR of 19.0% from 2022 to 2030. (leadschool11, 2023)

Before technology was applied to manage students, paper student management was a widely used method among many tutors. First, when the tutor wants to secure and protect personal information. When managing student information on paper, it is necessary to ensure that only those with specific authority, specifically tutors managing their students, can access and process the information. This ensures that students' personal information is not exposed or used improperly. The solution for tutors is to store student documents in safe places. A clear process should be maintained for checking and updating student information, while ensuring that only authorized persons have access to data. Second, data security and privacy are very important to every tutor. When managing student information on paper, tutors need to ensure compliance with data security and privacy regulations. Every tutor should not share with any third party. Security measures should be taken, and student documents protected from loss or unauthorized access. Although paper-based student management may be simple and easy to implement, it is still necessary to apply security measures and comply with legal regulations to ensure the safety and protection of students' personal information. .

When technology is applied to student management, the Excel method is a convenient and effective way to organize and store student information. When using Excel to manage students, it may involve sharing student information with other partners, authorities, and teachers. Before sharing student information with any legal partners, explicit consent from the parent or guardian is required. It is advisable to keep copies of consent records and ensure that information is only

disclosed for legal purposes and requirements. Next, tutors need to comply with regulations on online child protection: In using Excel to manage students, it is important to comply with regulations on online child protection. It is important to ensure that no sensitive or personal student information is exposed. Security measures should be created to prevent unauthorized access and keep student data in a secure environment. Instructions should be provided to staff about keeping student information private and not sharing it with any third parties without consent.

The trend has shifted from in-person platforms to online cloud-based systems that provide a complete set of tools to test the performance of every student. The software will empower the complete administrative and academic management of the school with a futuristic approach. Student management software is designed to organize and centralize student-related information, making it easily accessible to administrators and tutors Student management software provides individuals with ID and password password. Using that ID, they can easily track assignments, tuition status, grades, test results and parent information within seconds. (leadschool11, 2023)

Student management using an app is a modern and convenient way to organize and store information about students. First Student personal information: Using the app, you can store student personal information such as name, date of birth, address, phone number, and information about parents or guardians. Ensure that personal information is stored safely and securely. Apply security measures such as data encryption, user authentication, and access controls to prevent unauthorized access to student information. Second Timetable and scores: Using the app, you can create a timetable for each student, manage tests and record student scores. Create an easy-to-use interface to manage and update student schedules and grades. When developing applications, ensure that data is stored securely and editing is done only by authorized people. Third, Communication and notifications: With the app, you can send notifications and messages to parents or guardians about class schedules, important events, or students' personal information. Ensure that notifications are delivered effectively and securely. Control access to in-app notifications and messages to ensure privacy and data protection compliance. Finally, event and activity management: The app can also help you manage student

activities, events, and schedules. Create an easy-to-use interface to record and update event and activity information. Ensure that information about events and activities is only accessible to authorized people. Managing students with an app offers many significant benefits, such as convenience, ease of use, and can improve communication between teachers, parents, and students.

3.2 Technologies

3.2.1 Language

3.2.1.1 Java

Java is a programming language widely used for Android app development. It follows a class-based, object-oriented approach with syntax inspired by C++. Java aims to be simple, object-oriented, robust, secure, and high-level. While Java applications typically run on the JVM (Java Virtual Machine), Android uses its own virtual machine known as the Dalvik Virtual Machine (DVM), which is optimized for mobile devices (abhiandroid, 2024).

Popularity and Adoption

Java has been around for decades and has a massive community. It's widely adopted in enterprise applications, Android development, and more (Krysik, 2024).

Performance

Java's performance is generally good due to the Just-In-Time (JIT) compilation and optimization techniques of the JVM (Raj, 2024).

Ecosystem and Libraries

Java has a vast ecosystem with numerous libraries and frameworks available for various purposes, including Spring, Hibernate, and Android SDK (Mirzajanzadeh, 2024).

Platform Support

Java is platform-independent, thanks to the JVM, which means you can run Java code on any device with a JVM (Gupta, 2024).

3.2.1.2 Swift

Swift is a general-purpose programming language known for its approachability and power. It's modern, safe, and efficient, suitable for a wide range of applications from systems programming to mobile and desktop apps, as well as cloud services. Swift prioritizes safety by avoiding undefined behavior and making the obvious path the safest. It offers fast performance comparable to C-based languages while being developer-friendly. Swift is both easy to learn for

newcomers and powerful enough for large-scale applications, scaling according to project needs (swift, 2024).

Popularity and Adoption
Swift is primarily used for iOS and macOS app development (Tillu, 2023).

Performance
Swift is designed to be fast and efficient, with performance comparable to Objective-C. It benefits from being a compiled language (Bhatt, 2024).

Ecosystem and Libraries
Swift has a growing ecosystem, primarily focused on iOS/macOS development. There are many third-party libraries available via CocoaPods and Swift Package Manager (codementor, 2023).

Platform Support
Swift is mainly used for Apple platforms (iOS, macOS, watchOS, tvOS), although there are efforts to bring Swift to other platforms (swift, 32024).

3.2.1.3 Flutter

Using just one codebase, developers can create stunning natively built desktop, web, and mobile applications using Flutter, Google's portable UI toolkit. Flutter is free and open-source software that integrates with existing code and is utilized by developers and organizations globally (flutter, 2024).

Popularity and Adoption
Flutter has gained popularity relatively quickly, especially for mobile app development. Its community is growing, and it's becoming a popular choice for cross-platform development (Bui, 2021).

Performance
Flutter apps can achieve near-native performance because they are compiled to native machine code using the Dart compiler (Nayak, 2023).

Ecosystem and Libraries
Flutter has a rich set of customizable widgets and packages available via pub.dev. It also offers plugins for integrating with native code (Ahmed, 2024).

Platform Support

Flutter allows you to build cross-platform apps for iOS, Android, web, and desktop from a single codebase, making it highly versatile (logixbuilt, 2024).

3.2.2 Database

3.2.2.1 Firebase:

Firebase is a Backend-as-a-Service (Baas). It offers a range of tools and services to developers so they can create high-quality apps, expand their user base, and make money. It is built using Google's technical framework. A NoSQL database program, Firebase stores data in documents that resemble JSON.

- Database type:
  Firebase offers two types of databases: Firebase Realtime Database and Cloud Firestore, both are NoSQL databases serving different use cases and requirements (Lido, 2023).

- Real-time data synchronization:
  The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in real-time to every connected client. When you build cross-platform apps with our Apple platforms, Android, and JavaScript SDKs, all your clients share one Realtime Database instance and automatically receive updates with the newest data (firebase, 2023).

- Query language:
  Firebase offers two cloud-based, client-accessible document databases. We recommend new customers start with Cloud Firestore:

Cloud Firestore is the recommended enterprise-grade JSON-compatible document database, trusted by more than 250,000 developers. It's suitable for applications with rich data models requiring queryability, scalability, and high availability. It also offers low latency client synchronization and offline data access (firebase, 2023).

Realtime Database is the classic Firebase JSON database. It's suitable for applications with simple data models requiring simple lookups and low-latency synchronization with limited scalability (firebase, 2023).

- Security:

Firebase Security Rules stand between your data and malicious users. You can write simple or complex rules that protect your app's data to the level of granularity that your specific app requires (firebase, 2023).

3.2.2.2 SQLite

SQLite is an embedded, server-less relational database management system. It is an in-memory open-source library that needs no installation or configuration. It is also very practical because it is much smaller than other database management systems—less than 500kb in size.

- Database type:

SQLite only has four primitive data types: INTEGER, REAL, TEXT, and BLOB. APIs that return database values as an object will only ever return one of these four types. Additional .NET types are supported by Microsoft.Data.Sqlite, but values are ultimately coerced between these types and one of the four primitive types (microsoft, 2021). Data is stored in a single file, which reduces deployment complexity.

- Real-time data synchronization:

Does not support real-time data synchronization natively. Manual data synchronization and updates need to be handled.

- Query language:

SQLite understands most of the standard SQL language. But it does omit some features while at the same time adding a few features of its own. This document attempts to describe precisely what parts of the SQL language SQLite does and does not support. A list of SQL keywords is also provided. The SQL language syntax is described by syntax diagrams (sqlite, 2023).

- Security:

As SQLite databases are not encrypted or password-protected by default, ways of maintaining their security should be found. When an SQLite database is stored on the file system of your server, if it's stored in a location that is "web accessible" (whether this URL is announced or not) it is possible that this could be found, downloaded, and exploited (sqlite, 2023).

3.3 Methodologies

3.3.1 Waterfall:

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases (tutorialspoint, 2023).

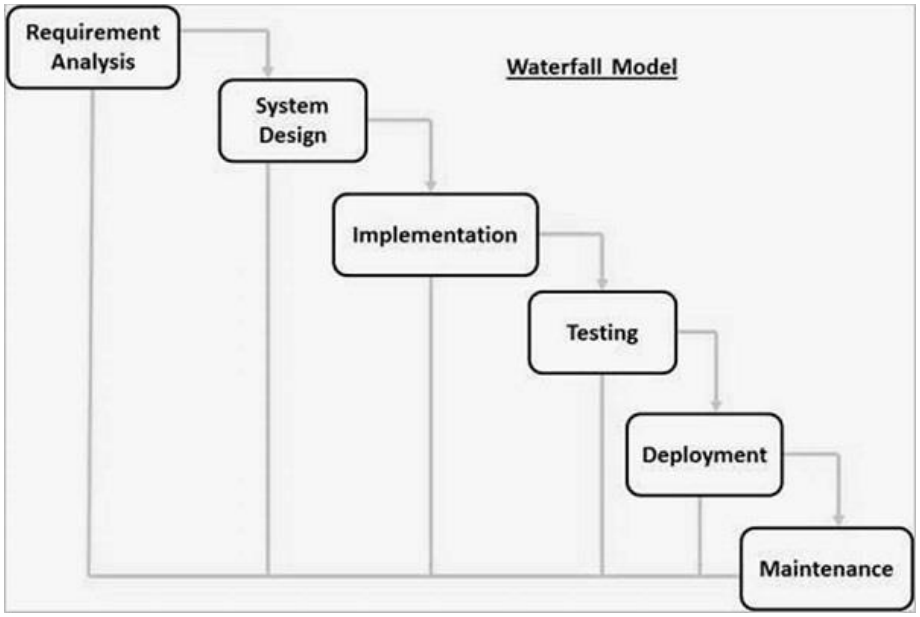


Figure 1: waterfall model

Working process:

Project phases (analysis, design, development, testing, deployment and Maintenance). All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap (tutorialspoint, 2023).

Customer interaction:

Minimal customer involvement: A waterfall project involves minimal customer interaction. This is primarily due to the fact that operations only start after the customer's requirements and objectives are clearly defined. The first meeting takes place before operations begin and the next when the project is in its final stages (Waseem, 2023).

Project scope:

The waterfall approach suits projects with stable requirements and a well-defined scope (usemotion, 2023). For small projects that have limited scope and do not require a high level of activity, the Waterfall model can be a good choice.

3.3.2 Agile:

Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In Agile, the tasks are divided to time boxes (small time frames) to deliver specific features for a release (tutorialspoint, 2023).

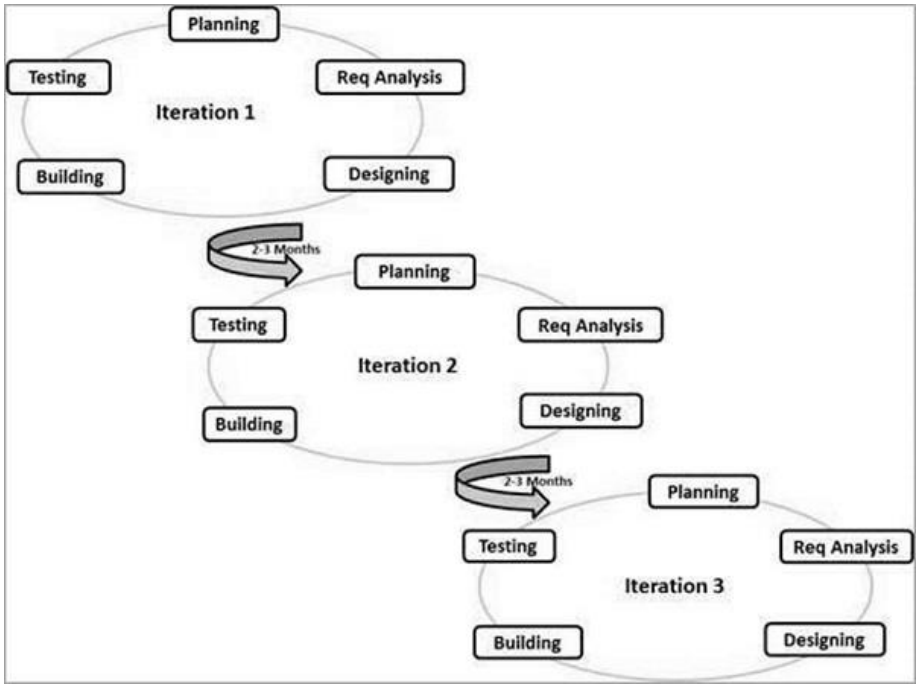


Figure 2: Agile model

Working process:

Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer (tutorialspoint, 2023). Use an agile model, breaking projects into short cycles called "sprints" and developing the product incrementally over each cycle.

Customer interaction:

Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction (tutorialspoint, 2023).

Project scope:

Agile Scope Management suggests that the scope of a project should not be considered fixed and unchangeable, as this may cause missed opportunities to increase business value. Unlike Waterfall management practices, Agile thinking views change as an opportunity rather than a problem (bvop, 2023). Agile is often favored in large and complex projects.

3.4 Choosing solutions

3.5 Technologies

3.5.1 Language

Criteria	Java (score: 5)	Swift (score: 5)	Flutter (score: 5)
Popularity and Adoption	Widely adopted for enterprise & Android (4/5)	Primarily used for iOS and macOS (3/5)	Rapidly gaining popularity for mobile apps (4/5)
Performance	Good due to JIT compilation & JVM (4/5)	Fast and efficient, comparable to Objective-C (4/5)	Near-native performance (5/5)
Ecosystem and Libraries	Vast ecosystem with Spring, Hibernate, Android SDK (5/5)	Growing ecosystem with CocoaPods & Swift Package Manager (4/5)	Rich set of widgets and packages via pub.dev (4/5)
Platform Support	Platform-independent via JVM (5/5)	Mainly Apple platforms (iOS, macOS, watchOS, tvOS) (3/5)	Cross-platform (iOS, Android, web, desktop (3/5)
Total:	19	14	16

Table 1: compare language

From the comparison table above, choosing Java for this project brings a balance of performance, scalability, community support, and rich ecosystem, making it a reliable choice.

3.5.2 Database

Criteria	Firebase (score: 5)	SQLite (score: 5)
Database Type	Offers two NoSQL databases: Realtime Database and Cloud Firestore (4/5).	Single relational database with limited data types (INTEGER, REAL, TEXT, BLOB) (3/5).
Real-time Data Sync	Real-time synchronization for connected clients across platforms (5/5).	Does not support native real-time synchronization (1/5).
Query Language	Cloud Firestore with rich query capabilities (4/5).	Supports most standard SQL with some omissions and additions (5/5).
Security	Firebase Security Rules for granular data protection (5/5).	Not encrypted or password-protected by default, requires additional security measures (4/5).
Total	18	13

Table 2: compare database

Based on the table above, Firebase scores higher due to its cloud-based infrastructure, real-time synchronization, rich query capabilities, better security features, and scalability options. SQLite, while lightweight and efficient for certain use cases, lacks many of these features.

3.6 Methodologies

Criteria	Waterfall (Score: 5)	Agile (Score: 5)
Working Process	Linear-sequential process where each phase must be completed before the next begins (5/5).	Iterative approach with working software delivered after each iteration (3/5).
Customer Interaction	Minimal customer involvement with defined requirements before project start and limited meetings (4/5).	Heavy customer interaction: requirements can evolve and change during development (2/5).
Project Scope	Best suited for projects with stable and well-defined scope. Limited flexibility for changes once development begins (3/5).	Scope is flexible and can change throughout the project. Adapts well to evolving requirements (2/5).
Flexibility	Low flexibility due to sequential nature; changes can be costly (3/5).	High flexibility with the ability to adapt to changes quickly (5/5).

Total:	15	12

Table 3: compare methodologies

Based on the table above, I choose Waterfall because: The waterfall model excels in structured environments, where requirements are clearly defined from the beginning and few changes are expected throughout the project lifecycle. It scores high in workflow and project scope for such situations. The Agile model is inherently flexible and adapts well to changing requirements, so it is suitable for complex and large-scale projects. It scores highly for versatility and complexity handling. However, due to its repetitive nature, it may require more frequent customer interactions, which can sometimes be a challenge.

4 Requirement Analysis

4.1 Similar applications

4.1.1 Edmodo:

Edmodo is a widely used flexible educational platform that helps teachers interact with students and implement lessons according to CCSS standards. With an easy-to-use interface, it facilitates communication and collaboration between teachers, students and parents. Edmodo also functions as a learning management system, providing online storage space for assignments and resources. Availability on multiple devices enables flexible learning and communication, supporting online learning and communication between teachers and students. Additionally, it also connects stakeholders in education, encouraging the exchange of ideas and resources. Edmodo not only improves the learning experience but also expands teachers' professional networks (Edmodo, 2024).

Edmodo offers a plethora of advantages that enhance the educational experience for both teachers and students. One of its standout features is team coordination, streamlining communication and collaboration among educators. The platform facilitates regular student check-ins, ensuring that no student feels left behind. It provides a safe and collaborative environment where teachers can share and store resources, fostering a culture of resource-sharing within the educational community. With engagement tools and assignment schedules,

Edmodo promotes active learning and keeps both teachers and students organized. Its seamless Google integration, class discussions, and ease of access further contribute to its appeal. Notably, its aesthetically pleasing interface, reminiscent of popular social media platforms, coupled with strong customer support and robust formative assessment capabilities, make Edmodo a comprehensive solution for modern education (Edmodo, 2024).

Despite its many strengths, Edmodo does come with some drawbacks that users have noted. A significant concern is the lack of control over personal messaging features, making it challenging to maintain appropriate teacher-student communication boundaries. Additionally, users have expressed a desire for the return of features like Snapshots for formative assessments and improvements in the user interface for a smoother experience. The connection with Google Education could be more seamless, and users have reported occasional technical glitches, especially on the mobile version. There have also been isolated incidents of student responses disappearing and persistent notification issues. While Edmodo's mobile version doesn't quite match up to its web counterpart in terms of functionality, and occasional performance slowdowns occur, these challenges don't overshadow its overall utility in the educational landscape (Edmodo, 2024).

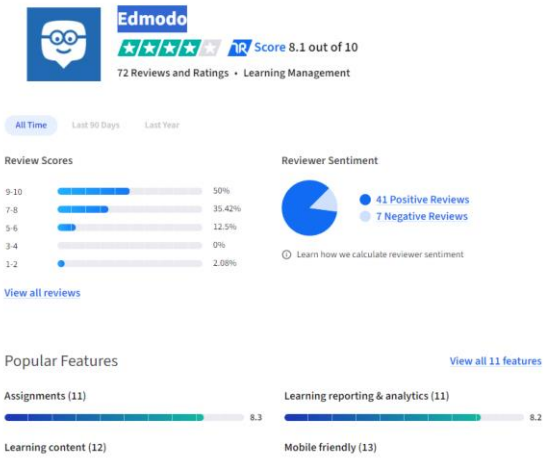


Figure 3: Edmodo

4.1.2 Canvas:

Canvas is a highly regarded learning management system adopted by universities, colleges, and schools for its user-friendly interface and effectiveness in managing both online and blended courses. It bridges communication between instructors and students, aiding in assignments, content sharing, and grade management. Beyond academics, Canvas serves HR for skill assessments, Student Services for club promotions, and libraries for information dissemination. Its mobile app enhances accessibility, and its frequent updates and robust features make it versatile for various course types. Overall, Canvas is essential for delivering quality education and promoting student engagement with its diverse features (canvas, 2024).

Canvas impresses users with its frequent updates, showcasing its commitment to improving the user experience and staying current. The mobile grading feature, Speed Grader, reduces grading time significantly, making it convenient for instructors to grade on the move. Additionally, Canvas's intuitive WYSIWYG editor simplifies content creation, allowing users to create engaging course materials easily, enhancing the learning experience for students (canvas, 2024).

While Canvas offers many features appreciated by users, there are areas of concern. Some find the navigation confusing and non-intuitive, making it challenging to find desired functionalities. Users also express dissatisfaction with the limited customization options for personalizing their experience. Additionally, there are concerns about the quality of customer support, with reports of delayed responses and availability issues (canvas, 2024).

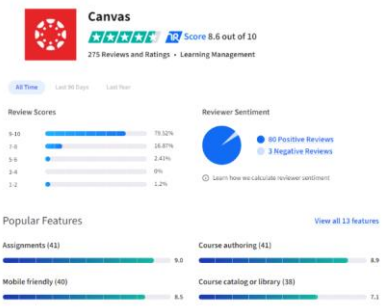


Figure 4: Canvas

4.1.3 Schoology

Schoology is a comprehensive Learning Management System popular among educators, students, and school districts for enhancing learning and communication. Teachers use it to share resources, post assignments, and assess students, while also coordinating school activities and fostering school culture. It facilitates easy communication between educators and families, manages events, assignments, and assessments, and encourages interactive class discussions. Students access course materials and assignments, benefiting from interactive learning. Parents receive regular updates on their child's progress. Schoology's user-friendly interface has led to its widespread adoption by school districts for addressing educational needs. Additionally, organizations use Schoology for professional development and training. Overall, Schoology simplifies the transition to digital classrooms, offering valuable tools for education and collaboration (Learning, 2024).

Schoology offers valuable integrations like Google Assignment App, which is especially beneficial for schools using Google and Chromebooks. It also integrates well with Microsoft OneDrive and Google Drive, facilitating seamless document sharing and assignments. Additionally, the platform provides options for assessment monitoring for schools and districts. The upcoming interface updates promise a cleaner look and an elementary version, aiming to enhance user experience. More question types, including labeling, have been introduced for assessments, broadening the scope of evaluation tools available to educators (Learning, 2024).

Despite its pros, Schoology has faced criticisms, primarily around unfulfilled promises regarding feature additions and improvements. Grading Google Drive Assignments can be cumbersome due to scrolling issues, and short answer/essay questions in assessments aren't readily visible for grading. Many teachers report that known pain points are never addressed by support, even after escalation. The focus of Schoology's roadmap appears to be more on integrating with other PowerSchool products rather than resolving existing platform issues. Additionally, the platform lacks some basic features like the ability to move and resize pop-up editing boxes, and settings and due dates for assessments are inconsistently located, leading to potential functionality issues (Learning, 2024).

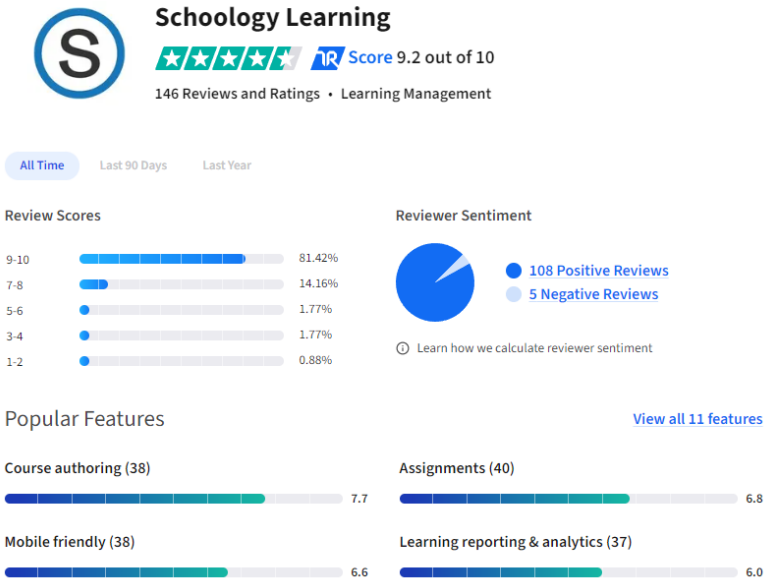


Figure 5: Schoology learning

4.1.4 Conclusion:

TutorKit Features :

User Roles:

- Tutor: Can register, log in, and update personal information.
- Student: Can register, log in, and update personal information. Can choose a tutor, view class schedules, grades, submit assignments, pay tuition, and take tests.

Tutor Features:

- Class Management: Manage class schedules.
- Student Management: Manage scores, assignments, and tuition for each student.
- Test Creation: Create tests for students.

Student Features:

- Tutor Selection: Choose a tutor.
- View Schedules: View class schedules.
- Grades: View grades.

- Assignments: Submit assignments.
- Tuition: Pay tuition.
- Tests: Take tests.

Personalized Matching: One of TutorKit's standout features is its personalized matching system. Students can browse through a list of tutors based on their expertise, availability. Once a student selects a tutor, the tutor can confirm the student, ensuring a mutual agreement before proceeding with lessons.

Comprehensive Tutor Management: Tutors using TutorKit can efficiently manage their class schedules, keep track of their students' progress, scores, assignments, and tuition. This comprehensive management system helps tutors stay organized and focused on delivering quality education.

Student-Centric Features: Students benefit from a range of features that enhance their learning journey. They can view their class schedules, grades, submit assignments, and even pay tuition through the platform. Additionally, students can take tests to assess their understanding and progress.

Secure and User-Friendly: TutorKit prioritizes the security of user information, ensuring that personal data and communication between tutors and students are protected. Its intuitive interface makes it easy for users of all ages to navigate and utilize its features effectively.

Flexible Learning: With TutorKit's online platform, both tutors and students can engage in flexible learning sessions. This flexibility enables students to learn at their own pace and allows tutors to manage their teaching hours according to their availability.

4.2 Requirement:

4.2.1 Project requirements:

Project information	Start: 01/02/2024
	End: 29/04/2024
Phase	Timeline
Requirements Gathering and Analysis	01/02/2024 - 10/02/2024
System Design and Architecture	11/02/2024 - 20/02/2024
Application development	21/02/2024 - 02/04/2024
Test and fix bug	03/04/2024 - 11/04/2024
Create documents	12/04/2024 – 29/04/2024

Table 4: project requirement

4.2.2 Solution requirement:

Item	Environment	Requirement
Developer	Hardware	Window 10 or more.
		Intel(R) Core(TM) i3-10105F CPU @ 3.70GHz 3.70 GHz
		8.00 GB RAM
	Software	Android Studio
		Java
	Database	Firebase
Tester	Hardware	Window 10 or more.
		Intel(R) Core(TM) i3-10105F CPU @ 3.70GHz 3.70 GHz
		8.00 GB RAM
	Software	Android Studio
		Java
	Database	Firebase
Product	Hardware	Window 10 or more.
		Intel(R) Core(TM) i3-10105F CPU @ 3.70GHz 3.70 GHz
		8.00 GB RAM
	Software	Android Studio
		Java
	Database	Firebase

Table 5: solution requirement

4.2.3 Business requirements:

No	Features	Estimation	Priority	Description
1	Tutor function			Tutor function
1.1	Register			
1.1.1	Register an account	1	AVERAGE	Tutor registers personal information, email address and password.
1.1.2	Verify mail	2	HIGH	Tutor must verify mail after successfully registering account.
1.2	Login/Logout			
1.2.1	Login to the application	1	HIGH	After successfully verifying the registered email. Tutor can use registered email and password to log in to the application.
1.2.2	Logout	1	AVERAGE	Tutor logouts their accounts from app.
1.2.3	Forgot password	2	HIGH	The application will send a link to the registered mail to enter a new password when you forget your login password.
1.3	Manage profile			Functions relate to tutor account
1.3.1	View	1	AVERAGE	Tutors view their accounts detail
1.3.2	Edit	1	AVERAGE	Tutors update their information.

1.3.3	Change mail	2	HIGH	Tutor enters the correct password to change the mail. If the change is successful, tutor must verify the new mail and login again.
1.3.4	Change password	2	HIGH	Tutor enters the correct password to change the password.
1.4	Accept students			Functions relate to tutor's student
1.4.1	View information's student	1	AVERAGE	Tutor can view some student information before accepting.
1.4.2	Accept	3	HIGH	Tutor accepts student to teach.
1.4.3	Cancel	1	AVERAGE	Tutor may not accept student.
1.4.4	View list tutor's student	1	HIGH	Tutor can view the list of students the tutor has accepted.
1.5	Manage schedules			Functions relate to schedule
1.5.1	Create	2	AVERAGE	Tutor creates schedule with student
1.5.2	Update	2	AVERAGE	Tutor updates schedule with student
1.5.3	Delete	2	AVERAGE	Tutor deletes schedule with student
1.5.4	View	1	AVERAGE	Tutor views all schedules for the week
1.6	Manage assignment			Functions relate to assignment
1.6.1	Create	2	AVERAGE	Tutor creates time to submit homework.
1.6.2	Update	2	AVERAGE	Tutor updates time to submit homework.
1.6.3	Delete	2	AVERAGE	Tutor deletes time to submit homework.
1.6.4	View	1	AVERAGE	Tutor views homework submission time and photos of students' submitted homework.
1.7	Exam (multiple choice)			Functions relate to exam
1.7.1	Create	5	HIGH	Tutor creates exam for student.
1.8	Manage grade			Functions relate to grade
1.8.1	Create	2	AVERAGE	Tutor creates grade from assignment.
1.8.2	Update	2	AVERAGE	Tutor updates grade from assignment.
1.8.3	Delete	2	AVERAGE	Tutor deletes grade from assignment.
1.8.4	View	1	AVERAGE	Tutor views grade from assignment and grade from exam.
1.9	Manage fee			Functions relate to fee
1.9.1	Create	2	AVERAGE	Tutor creates tuition invoices for students.
1.9.2	Update	2	AVERAGE	Tutor updates tuition invoices for students.
1.9.3	Delete	2	AVERAGE	Tutor deletes tuition invoices for students.
1.9.4	View	1	AVERAGE	Tutor views tuition invoices for students.
1.10	Chat			Functions relate to chat
1.10.1	Chat with student	3	HIGH	Tutors can only chat with their students
1.11	Call			Functions relate to call
1.10.1	Call with student	1	AVERAGE	Tutors can only call with their students
2	Student function			Student function

1.1	Register			
1.1.1	Register an account	1	AVERAGE	Student registers personal information, email address and password.
1.1.2	Verify mail	2	HIGH	Student must verify mail after successfully registering account.
1.2	Login/Logout			
1.2.1	Login to the application	1	HIGH	After successfully verifying the registered email. Student can use registered email and password to log in to the application.
1.2.2	Logout	1	AVERAGE	Student logouts their accounts from app.
1.2.3	Forgot password	2	HIGH	The application will send a link to the registered mail to enter a new password when you forget your login password.
1.3	Manage profile			Functions relate to tutor account
1.3.1	View	1	AVERAGE	Student view their accounts detail
1.3.2	Edit	1	HIGH	Student update their information.
1.3.3	Change mail	2	HIGH	Student enters the correct password to change the mail. If the change is successful, tutor must verify the new mail and login again.
1.3.4	Change password	2	HIGH	Student enters the correct password to change the password.
1.4	choose Tutor			Functions relate to student' tutor
1.4.1	View information's tutor	1	AVERAGE	Student can view some tutor information before choosing.
1.4.2	Choose	1	HIGH	Student chooses tutor.
1.4.3	View list student's tutor	2	HIGH	Student can view the list of Tutor the student has choosed.
1.5	View schedules			Functions relate to schedule
1.5.1	View	1	AVERAGE	Student views all schedules for the week
1.6	Submit assignment (image)			Functions relate to assignment
1.6.1	Submit	2	AVERAGE	Student submit assignment.
1.6.2	Delete	1	AVERAGE	Student cancel submission assignment.
1.6.3	View	1	AVERAGE	Student views assignment submitted.
1.7	Exam (multiple choice)			Functions relate to exam
1.7.1	Take an exam	5	HIGH	Student does exam.
1.8	View grade			Functions relate to grade
1.8.1	View	1	AVERAGE	Student views grade from assignment and grade from exam.
1.9	Manage fee			Functions relate to fee
1.9.1	Pay fee	3	HIGH	Student pays the tuition.

1.9.2	View	1	AVERAGE	Student views tuition invoices.
1.10	Chat			Functions relate to chat
1.10.1	Chat with tutor	3	HIGH	Student can only chat with their tutors
1.11	Call			Functions relate to call
1.10.1	Call with tutor	1	AVERAGE	Student can only call with their tutors
Total		92.0		

Table 6: Business requirements

4.2.4 Non-requirement:

Non-requirement	Description
Performance	Loading time <=5s
Security	Password >= 6 Password encryption Validate user input Authorization user Authentication user
UI/UX	Application element design should be unified (font-family, font size, button style, color, etc.) The application should be responsive and accessible on all types of devices smartphones with a user-friendly interface that adapts to different screen sizes.

Table 7: non-requirement

4.2.5 Stakeholder requirements:

Item	Requirement
Tutor	Manage schedule, grade, assignment, tuition Student. Create Exam View profile and update own profile.
Student	View schedule, grade, assignment, tuition Student. Take an Exam Submit assignment Pay tuition View profile and update own profile.
Developers	The IT team needs to guarantee the website's security, reliability, and scalability. Additionally, they should monitor performance, address bugs, and rectify errors.

Table 8: stakeholder requirement

5 Software design

5.1 Use case diagram:

Figure 6: use case diagram

5.2 ERD:

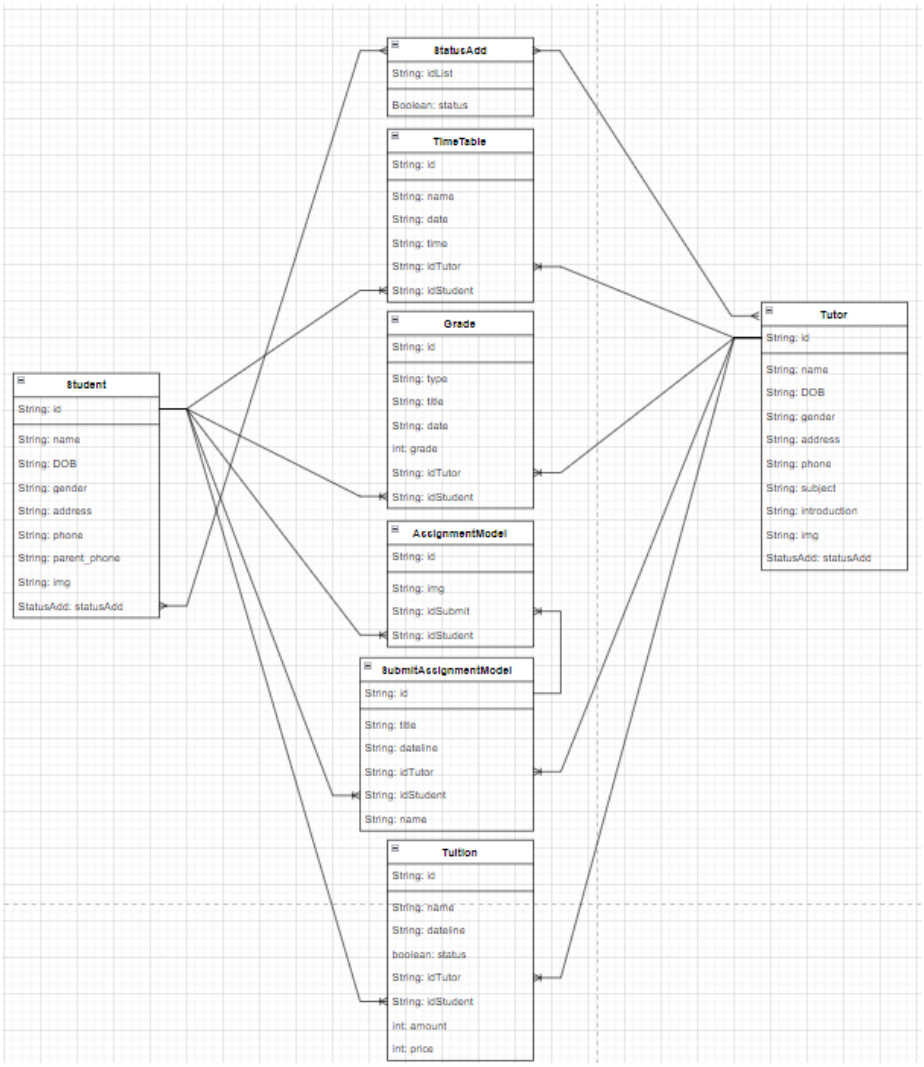


Figure 7: ERD

5.3 Flowchart:

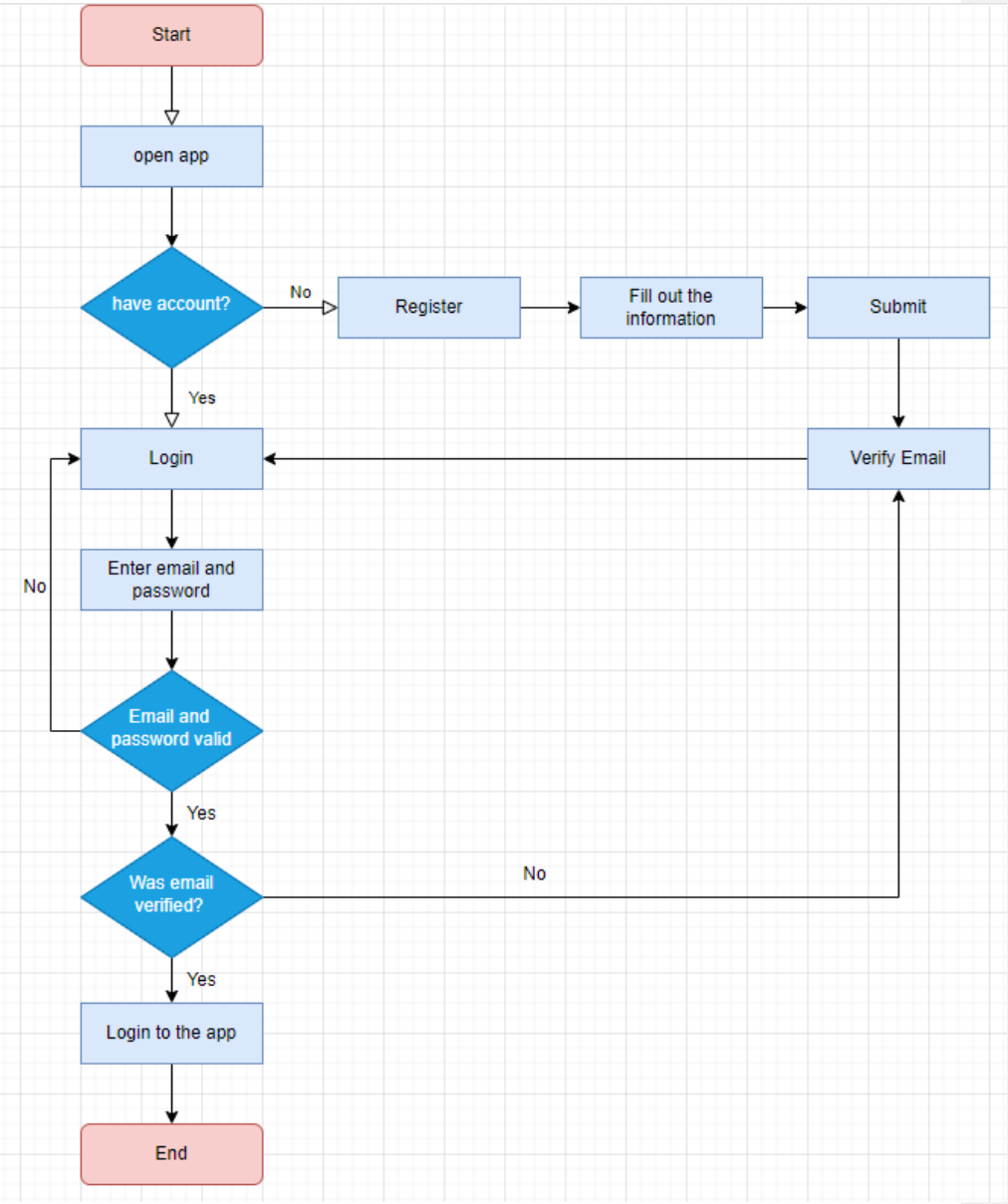


Figure 8: flowchart login

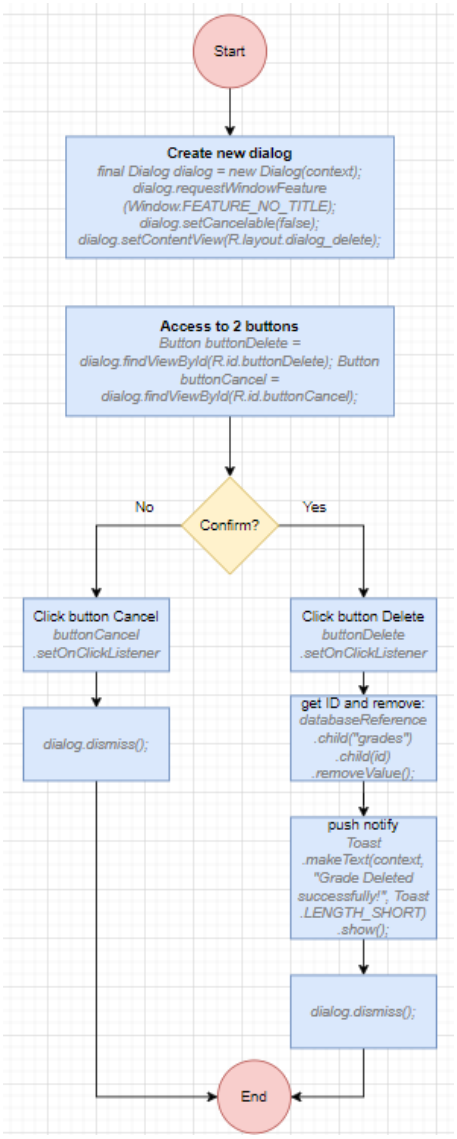


Figure 9: flowchart create

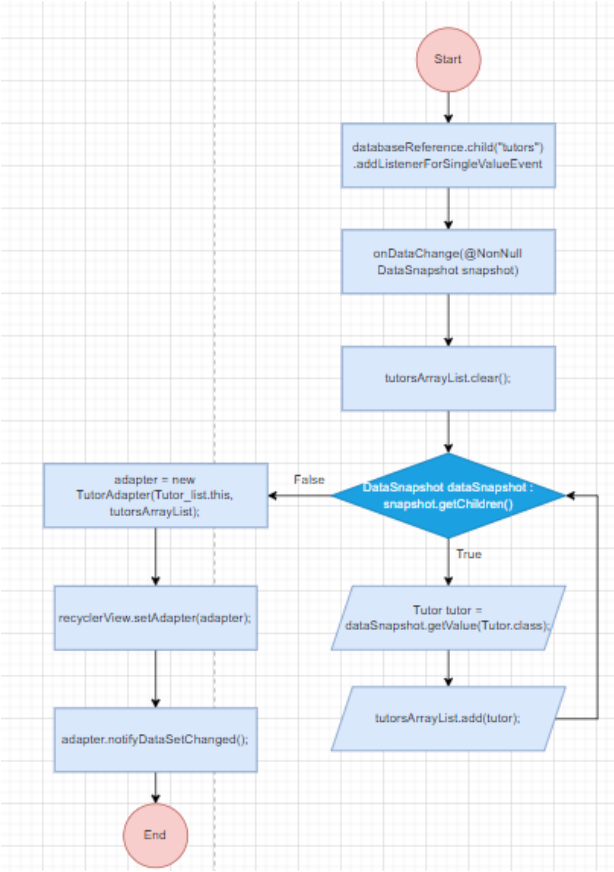


Figure 10: flowchart showListTutor

5.4 Activity diagram

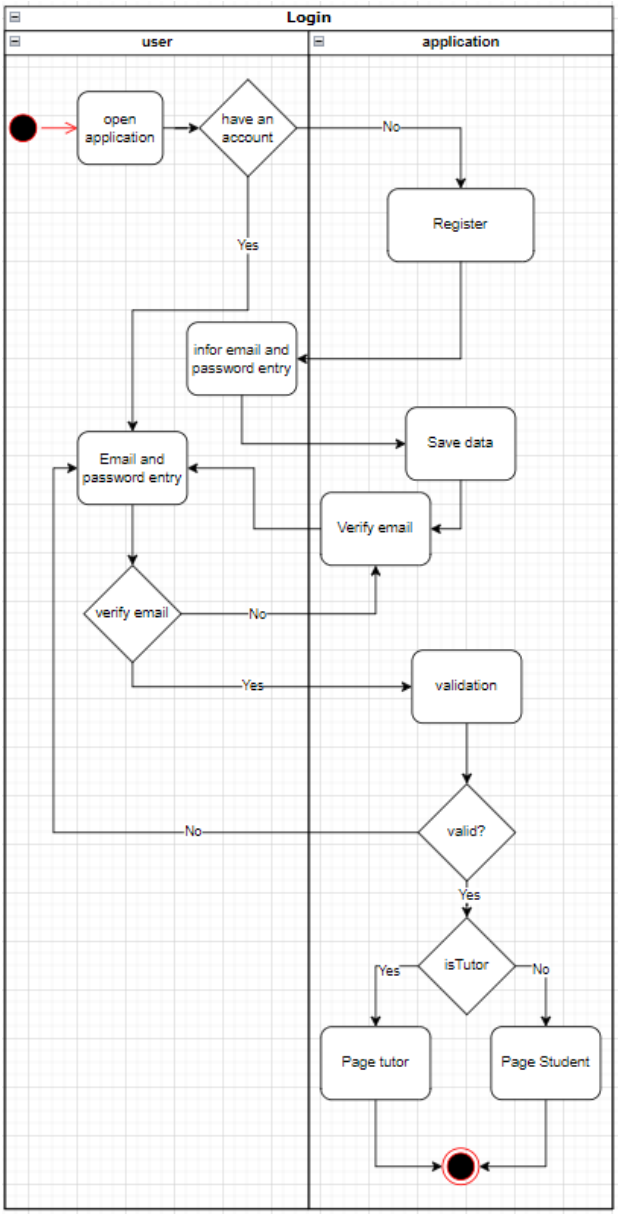


Figure 11: activity login

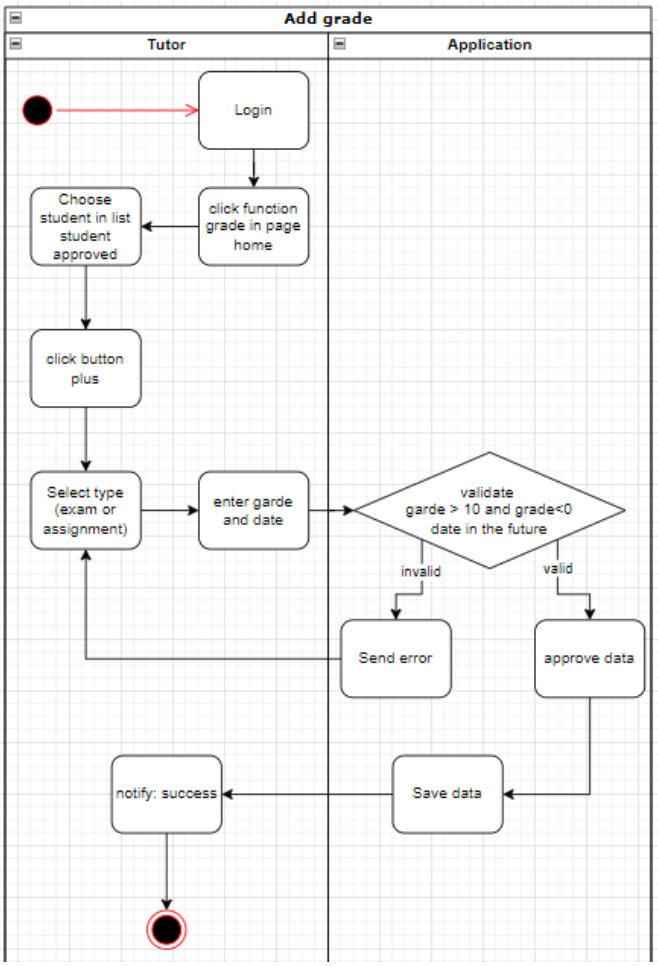


Figure 12: activity add grade

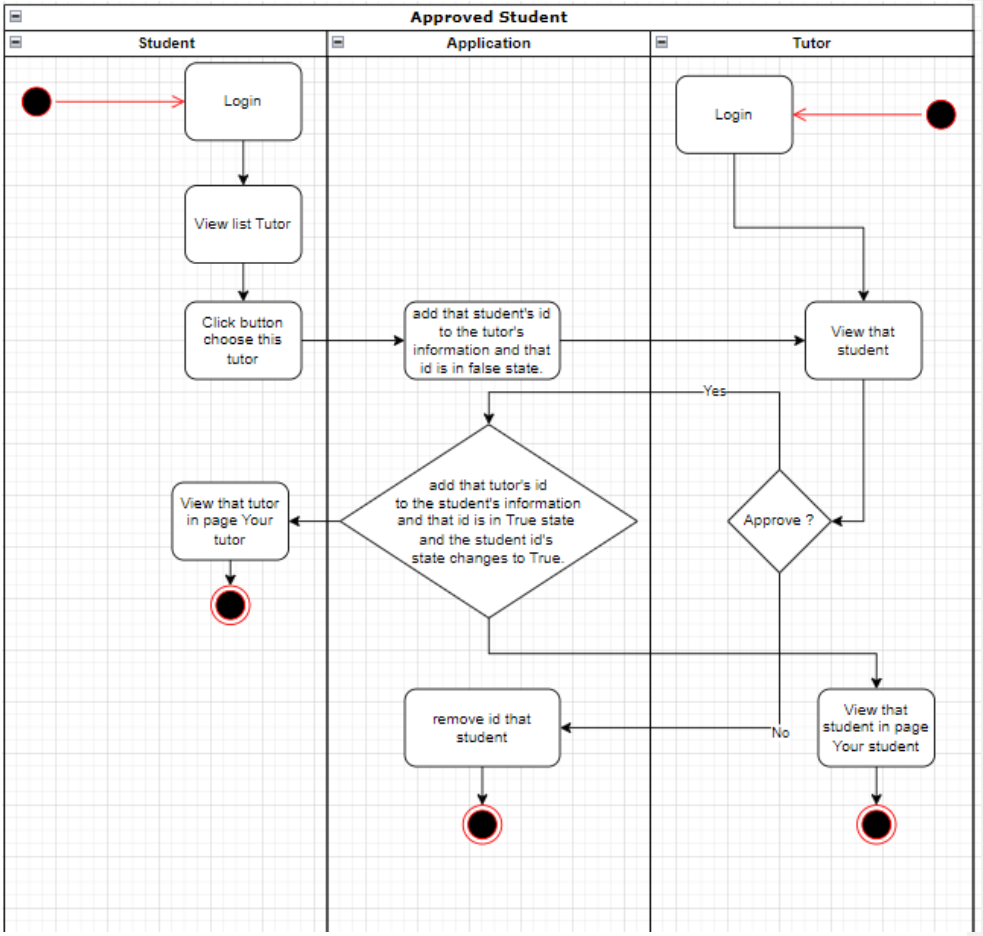


Figure 13: activity approved student

5.5 Wireframe:

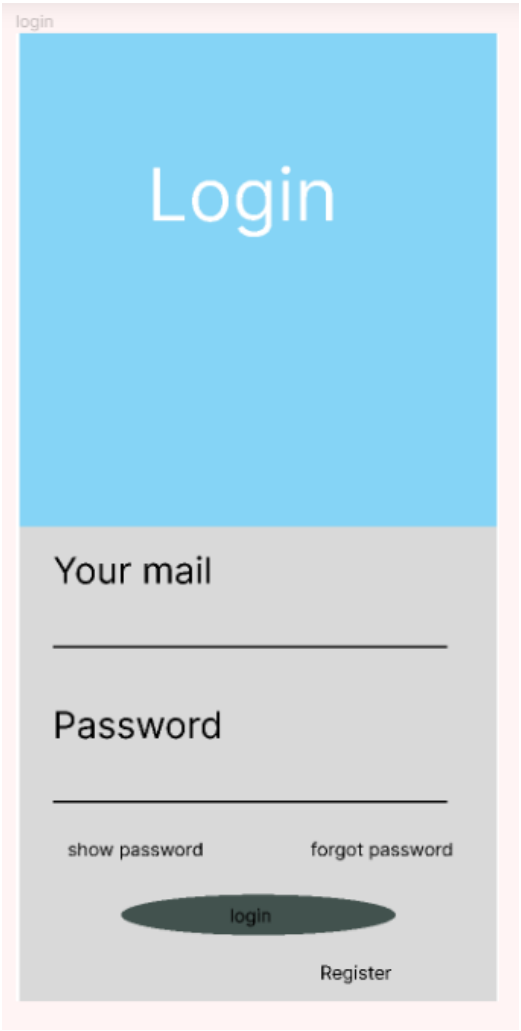


Figure 14: wireframe login

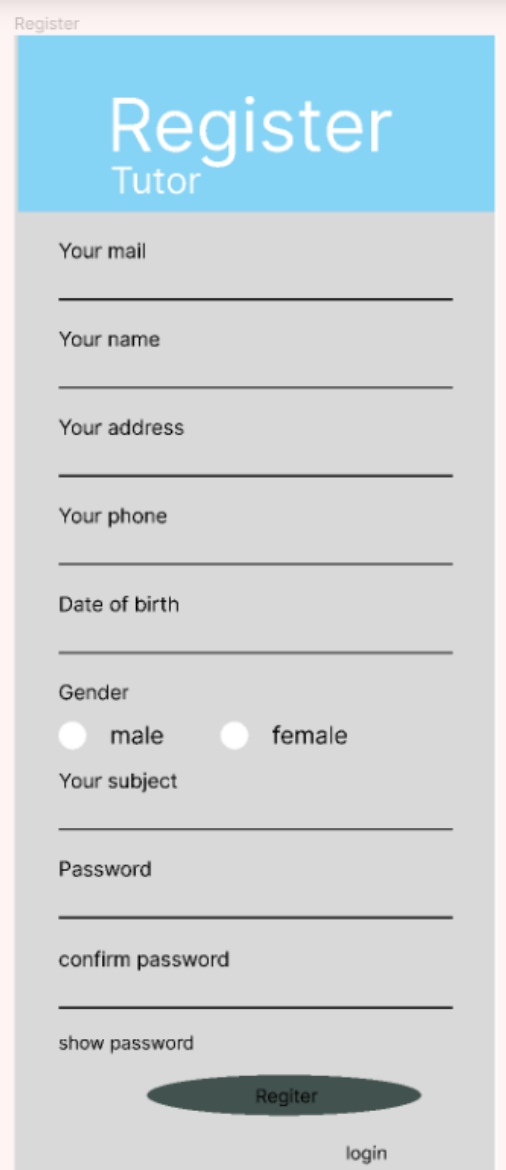


Figure 15: wireframe register tutor

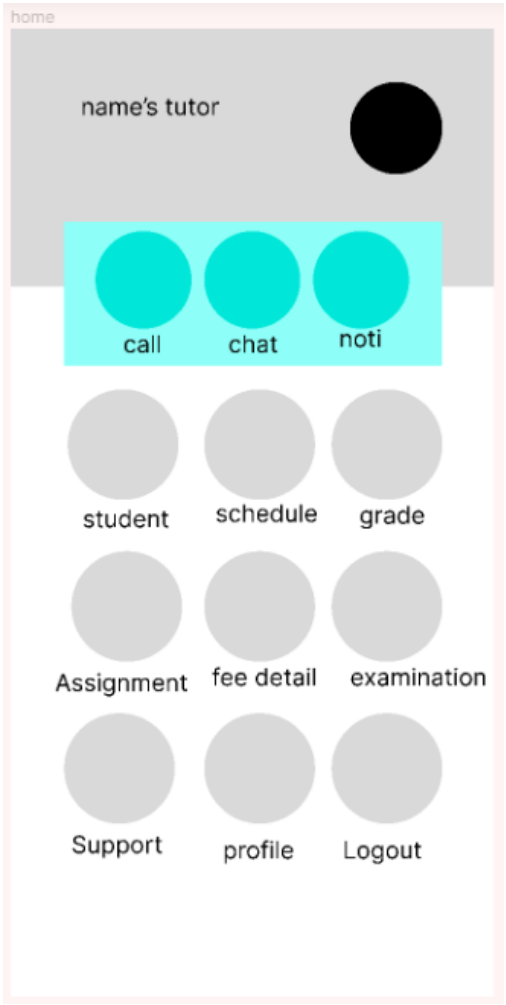


Figure 16: wireframe home tutor

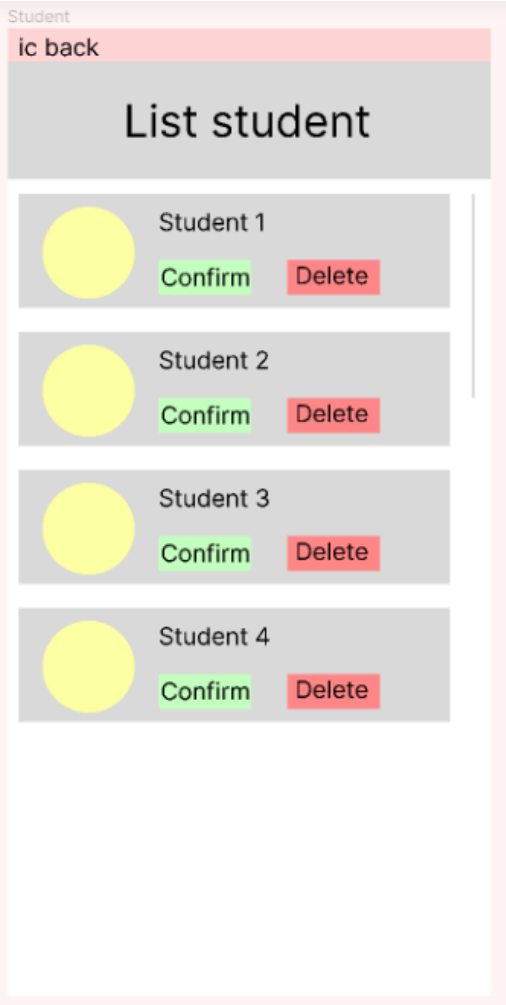


Figure 17: wireframe list Student

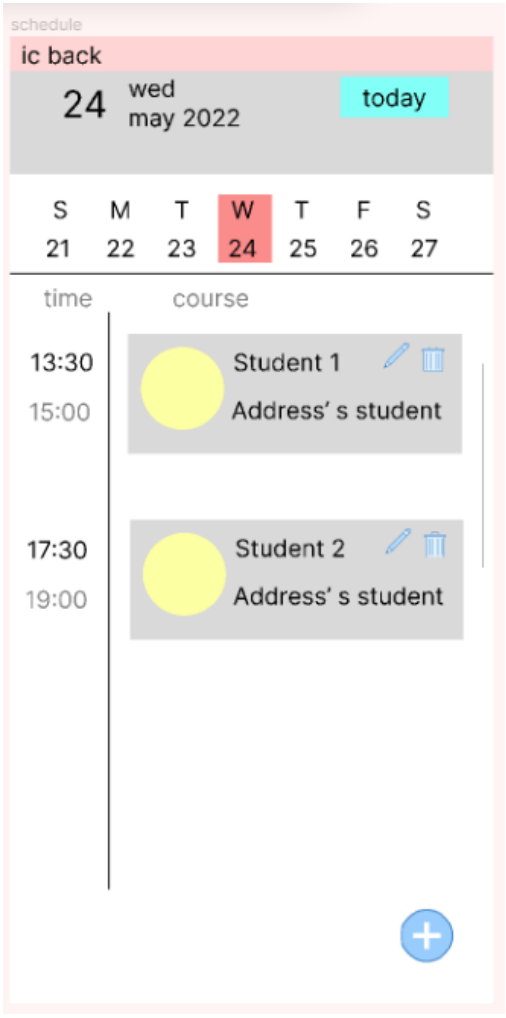


Figure 18: wireframe schedule

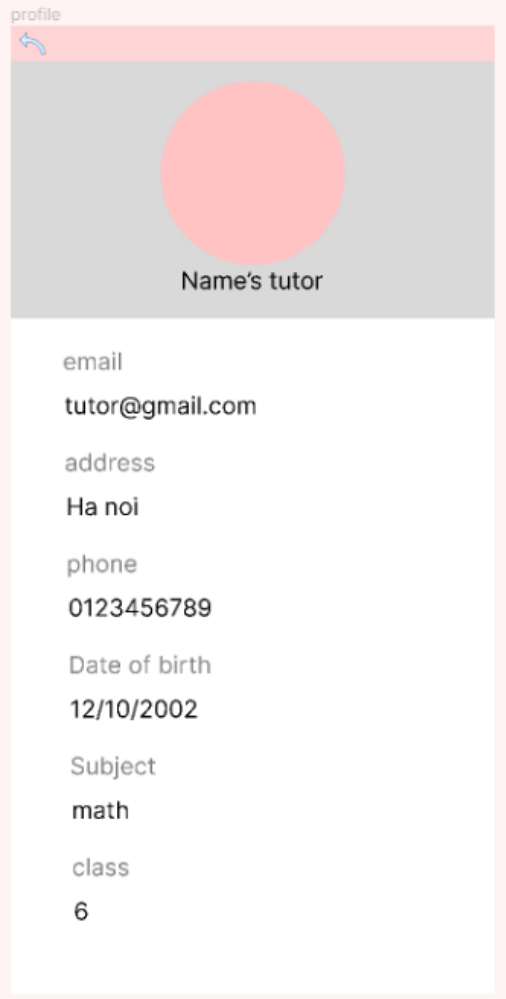


Figure 19: wireframe profile

Figure 20: wireframe get support

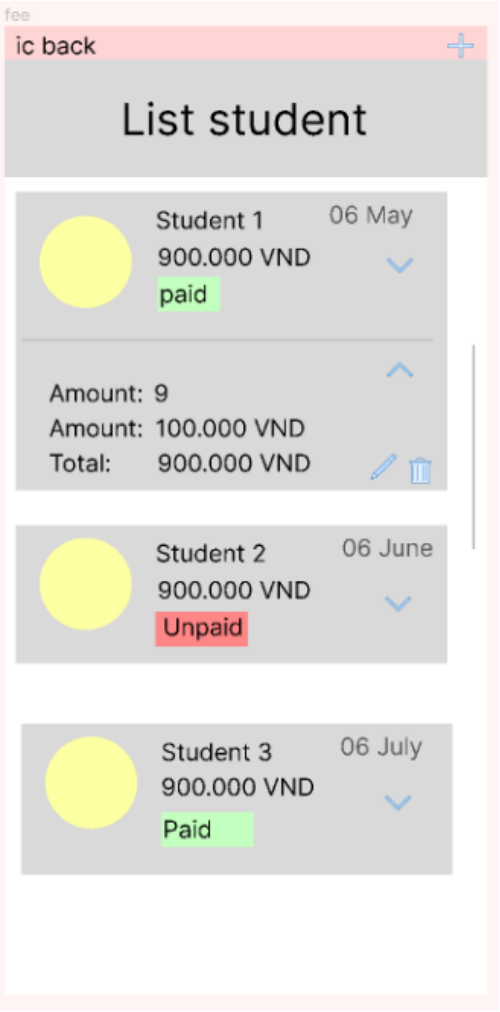


Figure 21: wireframe list Tuition

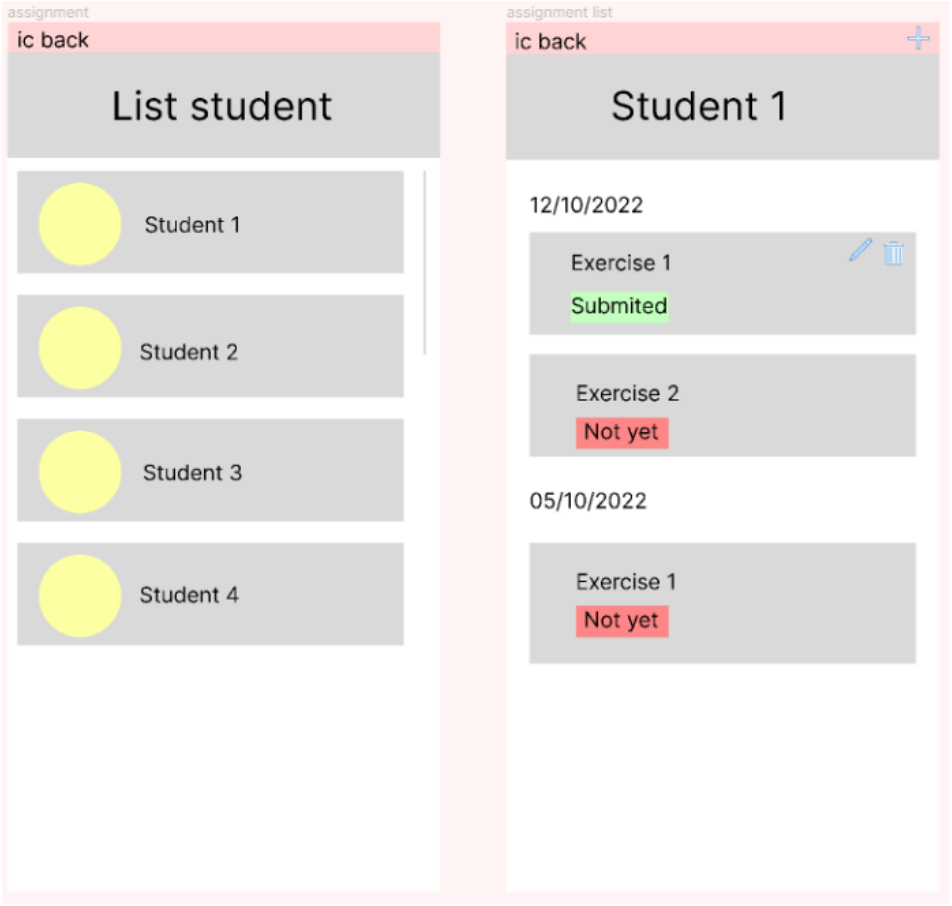


Figure 22: wireframe list assignment

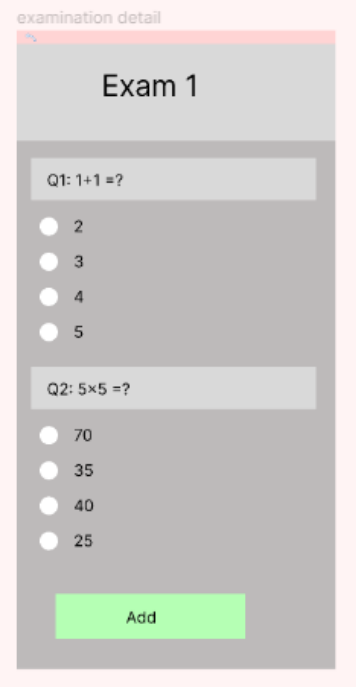


Figure 23: wireframe exam

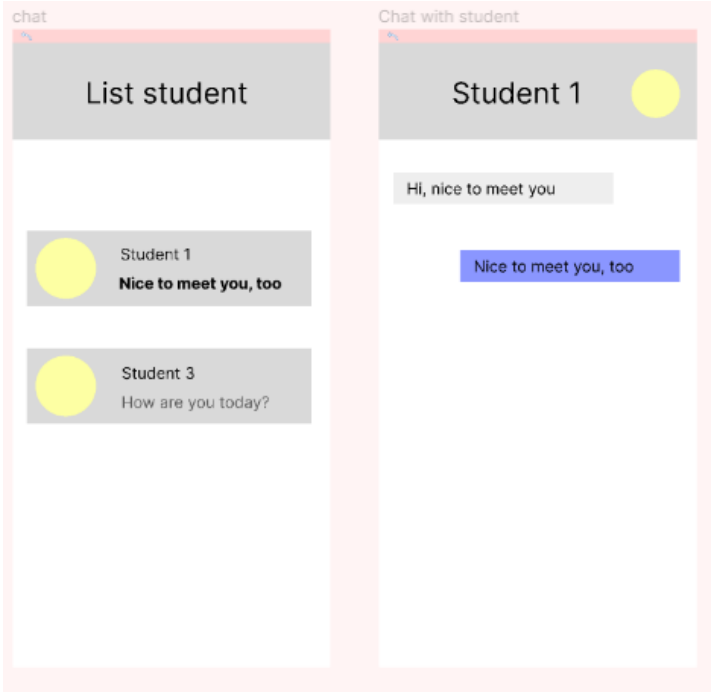


Figure 24: wireframe chat

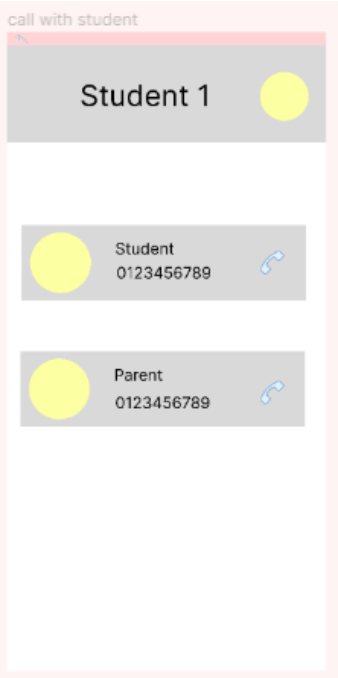


Figure 25: wireframe call

Figure 26: wireframe register Student.

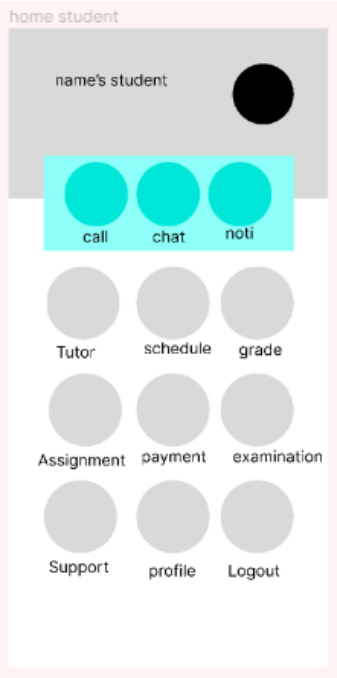


Figure 27: wireframe home student

5.6 Sitemap:

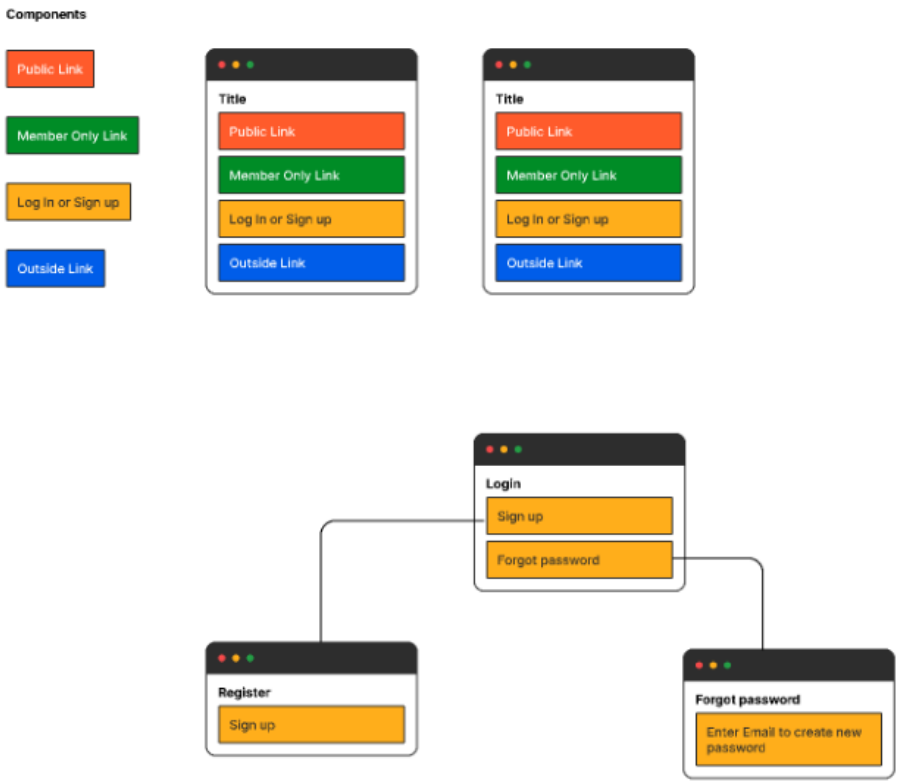


Figure 28: sitemap login

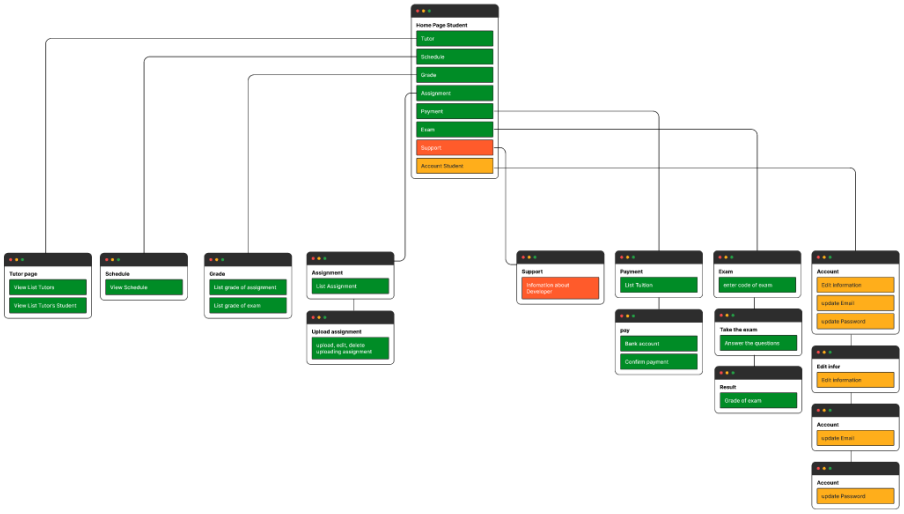


Figure 29: sitemap student

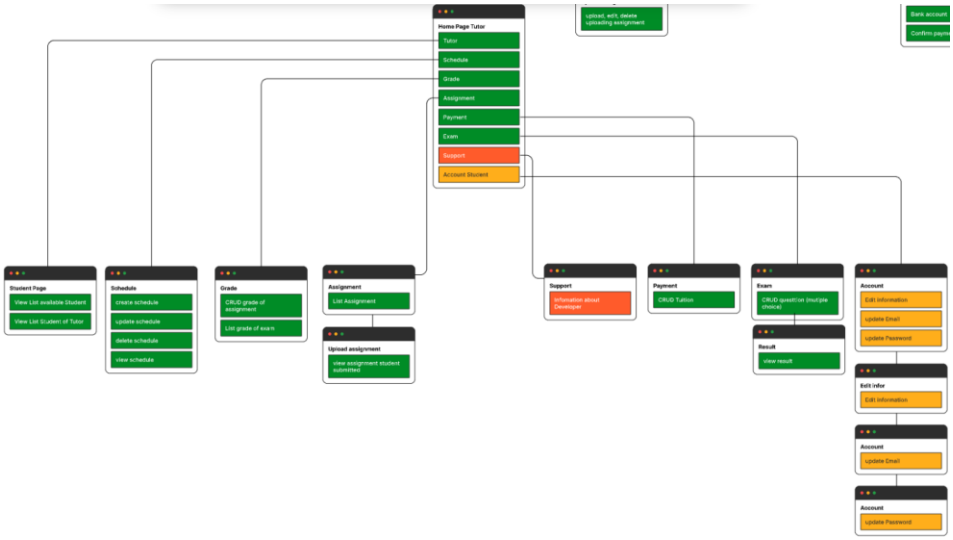


Figure 30: sitemap Tutor

6 Implementation

6.1 Development tools

I used many tools to complete this application. Those tools are:

Commented [H1]: Not done

6.1.1 Android Studio

Android Studio is the go-to IDE for Android app development, leveraging IntelliJ IDEA's powerful tools. It features a flexible Gradle-based build system, a fast emulator for testing, and a unified environment for developing across all Android devices. Live Edit enables real-time UI updates, while code templates and GitHub integration expedite development. The IDE also offers robust testing tools, Lint for quality checks, and supports C++ and NDK. Plus, built-in Google Cloud Platform support simplifies cloud integration for services like Google Cloud Messaging and App Engine (developer, 2024).

6.1.2 LDPlayer 9

LDPlayer 9 is a PC emulator that brings Android 9 to your desktop, offering enhanced performance for gaming and apps. With its user-friendly interface, LDPlayer 9 is easy to navigate and customize, even if you're new to emulators. It comes with the Google Play Store pre-installed and supports alternative stores like Uptodown. You can map controls for mouse and keyboard, making gaming more intuitive, and it also works with physical game controllers. The emulator offers extensive settings to optimize your gaming experience, from resource allocation to virtual machine configurations. Overall, LDPlayer 9 delivers a smooth and high-quality emulation experience, making it a great choice for gamers wanting to play Android games on PC (9, 2024).

6.1.3 SamSung Galaxy J2 Prime

This is a mobile device using the Android operating system. I use this to run the application. The device is connected to the computer via a cable.

6.1.4 Trello:

Trello is a visual work management tool designed to streamline team collaboration and project organization. It offers a flexible platform that can adapt to any project, whether you're starting something new or organizing existing work. With its intuitive interface, Trello simplifies and standardizes work processes, making it easy for teams to manage tasks, set priorities, and track progress. Despite its user-friendly design, Trello is powerful enough to handle even the most complex projects. As you start your first project on Trello, you'll find it to be a versatile tool that promotes productivity and teamwork (trello, 2024).

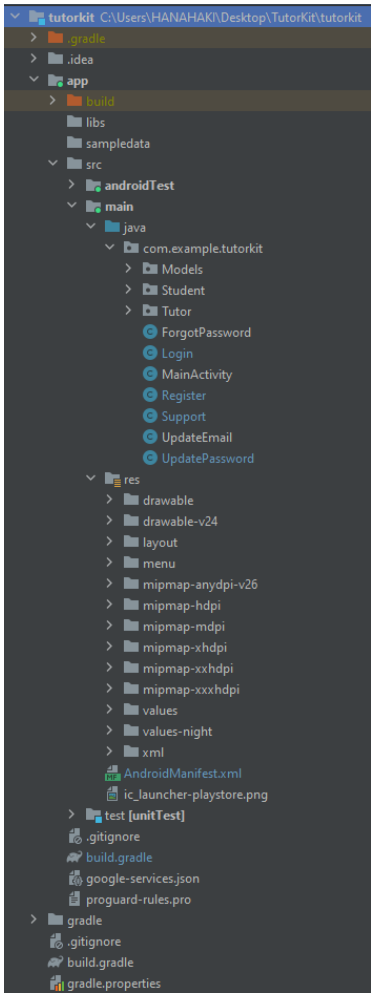


Figure 31: the project structure

6.2 Project Structure:

The project structure is shown in the figure below. Role of each folder:

+ Models: Stores the model source code of the entire application.

+ Student: Stores Student's functional source code in the application.

+ Tutor: Stores Tutor's functional source code in the application.

+ The remaining class in "com.example.tutorkit": Stores the source code for common features of both Student and Tutor.

+ Layout: Stores the user interface source code of the entire application.

+ Drawable: Stores images, icons,... used in the interface

6.3 Code snippets of important features:

6.3.1 Approve Student:

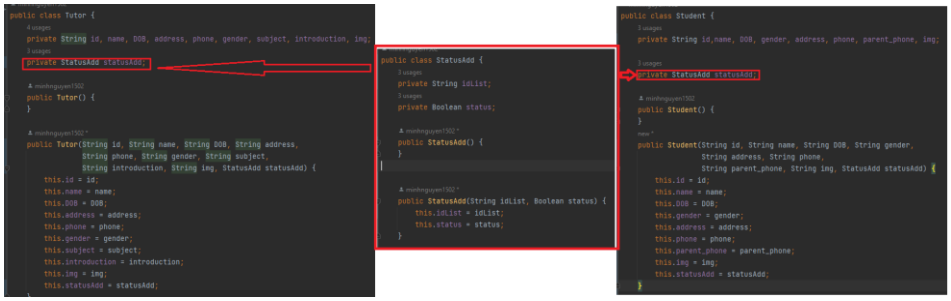


Figure 32: StatusAdd

```
try {
    // Check if the tutor ID exists in the Student
    DatabaseReference studentRef = FirebaseDatabase.getInstance().getReference( path: "Student")
            .child(FirebaseAuth.getInstance().getUid()).child( pathString: "IdTutors").child(tutorId);
    studentRef.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            StatusAdd statusAdd = dataSnapshot.getValue(StatusAdd.class);
            if (statusAdd != null) {
                choose.setVisibility(View.GONE);
            } else {
                choose.setOnClickListener(new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        FirebaseDatabase.getInstance().getReference( path: "tutors")
                                .child(tutorId)
                                .child( pathString: "IdStudent").child(FirebaseAuth.getInstance().getUid())
                                .setValue(new StatusAdd(FirebaseAuth.getInstance().getUid(),  status: false));
                        Toast.makeText( context: ViewProfileTutor.this,  text: "I liked this tutor", Toast.LENGTH_SHORT).show();
                    }
                });
            }
        }


        @Override
        public void onCancelled(@NonNull DatabaseError databaseError) {
            Toast.makeText( context: ViewProfileTutor.this,  text: "Failed to load student status", Toast.LENGTH_SHORT).show();
        }
    });
}catch (Exception e){
    e.getMessage();
}
```

Figure 33: button Choose Tutor

Try-Catch Block: The code begins with a try block to catch any exceptions that might occur during its execution.

DatabaseReference studentRef is initialized to reference the "Student" table in the Firebase Realtime Database under the current user's UID. Within "Student", it further drills down to "IdTutors" and then to a specific tutorId. If statusAdd (Figure 4: StatusAdd) is not null, it means that the tutor with tutorId exists in the student's "IdTutors" list. The button choose view's visibility is set to View.GONE to hide it. If statusAdd is null, it means the tutor hasn't been added to the student's list yet. An OnClickListener is set for the choose view. When select is clicked, it adds the current student's ID (FirebaseAuth.getInstance().getUid()) to the "tutors" table in the newly generated StatusAdd object and the status is false.

Catch Block: If any exception occurs in the try block, it's caught in the catch block.

```
holder.confirm.setOnClickListener(new View.OnClickListener() {
    minhnguyen1502 *
    @Override
    public void onClick(View view) {
        FirebaseDatabase.getInstance().getReference( path: "Student")
                .child(students.getId())
                .child( pathString: "IdTutors").child(FirebaseAuth.getInstance().getUid())
                .setValue(new StatusAdd(FirebaseAuth.getInstance().getUid(),  status: true));

        FirebaseDatabase.getInstance().getReference( path: "tutors")
                .child(FirebaseAuth.getInstance().getUid())
                .child( pathString: "IdStudent").child(students.getId())
                .setValue(new StatusAdd(students.getId(),  status: true));

        Toast.makeText(context,  text: "I choose this student", Toast.LENGTH_SHORT).show();
    }
});
 minhnguyen1502 *
holder.cancel.setOnClickListener(new View.OnClickListener() {
    minhnguyen1502 *
    @Override
    public void onClick(View view) {
        FirebaseDatabase.getInstance().getReference( path: "tutors")
                .child(FirebaseAuth.getInstance().getUid())
                .child( pathString: "IdStudent").child(students.getId())
                .removeValue();
        Toast.makeText(context,  text: "cancel", Toast.LENGTH_SHORT).show();
    }
});
```

Figure 34: confirm Student

When clicking the confirmation button:

It updates "IdTutors" into a new StatusAdd Object that takes the current user's ID (FirebaseAuth.getInstance().getUid()) and changes the id's state to true, which is then stored in the "Students" table.

At the same time it updates the status of "IdStudent" (StatusAdd(students.getId(), true) in the "tutors" table to true and will be stored.

When clicking the cancel button:

It deletes the specific student's ID entry in the "IdStudent" of the current user's UID in the "tutor" table.

```
private void showListStudents() {
    FirebaseDatabase.getInstance().getReference( path: "tutors")
            .child(FirebaseAuth.getInstance().getUid())
            .child( pathString: "IdStudent")
            minhnguyen1502 *
            .addListenerForSingleValueEvent(new ValueEventListener() {
        minhnguyen1502 *
        @SuppressLint("NotifyDataSetChanged")
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            idStudent.clear();
            for (DataSnapshot dataSnapshot : snapshot.getChildren()) {
                StatusAdd statusAdd = dataSnapshot.getValue(StatusAdd.class);
                try {
                    if (!statusAdd.getStatus()) {
                        idStudent.add(statusAdd.getIdList());
                    }
                }catch (Exception e){
                    Log.e( tag: "TAG",  msg: "onDataChange: "+e.getMessage() );
                }
            }
            if (idStudent.size()>0){
                for (int i =0; i<idStudent.size();i++){
                    databaseReference.child( pathString: "Student").child(idStudent.get(i))
                            minhnguyen1502 *
                            .addListenerForSingleValueEvent(new ValueEventListener() {
                        minhnguyen1502
                        @Override
                        public void onDataChange(@NonNull DataSnapshot snapshot) {
                            studentArrayList.add(snapshot.getValue(Student.class));
                            adapter.notifyDataSetChanged();
                        }
                        minhnguyen1502
                        @Override
                        public void onCancelled(@NonNull DatabaseError error) {

                        }
                    });
                }
            }else {
            }
        }
```

Figure 35: function showListStudent()

The purpose of method showListStudent() is fetching a list of students associated with a tutor from Firebase Realtime Database. *"FirebaseDatabase.getInstance().getReference("tutors").child(FirebaseAuth.getInstance().getUid()).child("IdStudent")".* This code initializes a reference to the Firebase Realtime Database, specifically pointing to the tutors table, then to the current user's (FirebaseAuth.getInstance().getUid()) table under tutors, and finally to the IdStudent child node.

I use for loop to iterate through each child IdStudent to retrieve StatusAdd objects. And I check the Status attribute of each StatusAdd object. If the status is false, it adds the corresponding IdList to the idStudent ArrayList.

The following if statement block has the following purpose: Checking if idStudent ArrayList has any entries. And Fetch student data via command block: *"databaseReference.child("Student").child(idStudent.get(i)).addListenerForSingleValueEvent(new ValueEventListener() { ... }"*. If there are entries in idStudent, it fetches the corresponding student data from the Student table in the database.

Final, notify the adapter that the data has changed so that it can refresh the RecyclerView or ListView displaying the student data.

6.3.2 Using firebaseAuth

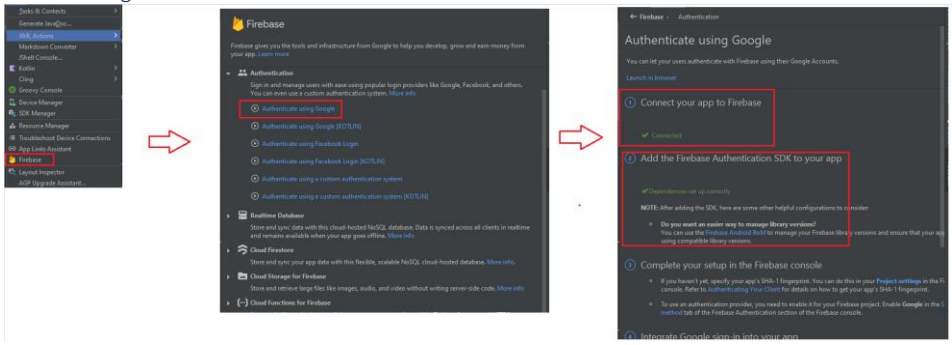


Figure 36: using firebase in project

Click in button tool in navbar and choose firebase and connect to Firebase next Add the Firebase Authentication dependency.

```
implementation 'androidx.appcompat:appcompat:1.6.1'
implementation 'com.google.android.material:material:1.5.0'
implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
implementation 'com.google.firebase:firebase-database:20.1.0'
implementation 'com.google.firebase:firebase-auth:21.1.0'
implementation 'com.google.firebase:firebase-storage:20.1.0'
testImplementation 'junit:junit:4.13.2'
androidTestImplementation 'androidx.test.ext:junit:1.1.5'
androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
```

Figure 37: SDK of firebase

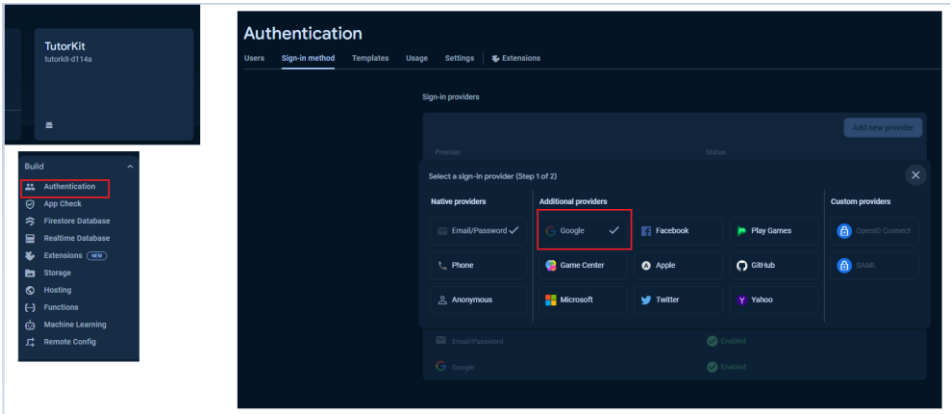


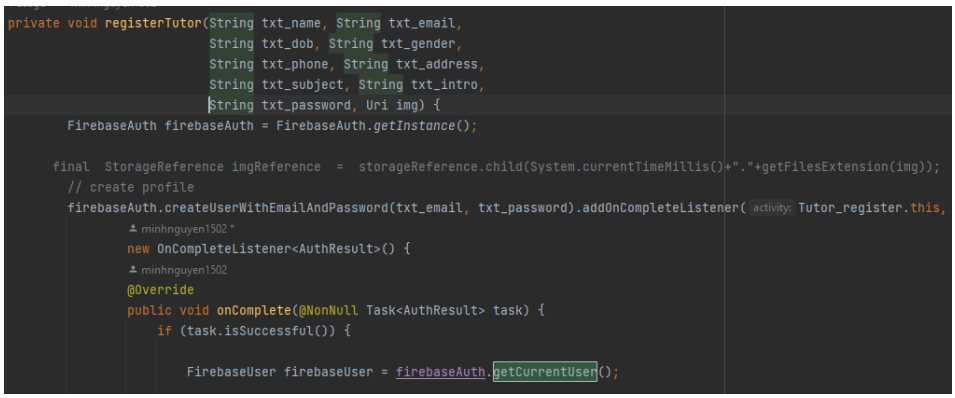
Figure 38: set up in Firebase

Go to the Firebase Console and create a new project or select an existing one. And in this project I using Email/ password with Google.

```
public class Login extends AppCompatActivity {

    2 usages
    TextView register;

    10 usages
    private EditText edt_email, edt_password;
    3 usages
    FirebaseAuth firebaseAuth;
    minhnguyen1502 *
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        register = findViewById(R.id.txt_register);
        edt_password = findViewById(R.id.edt_password);
        edt_email = findViewById(R.id.edt_email);
        firebaseAuth = FirebaseAuth.getInstance();
```

Figure 39: initialize firebaseAuth

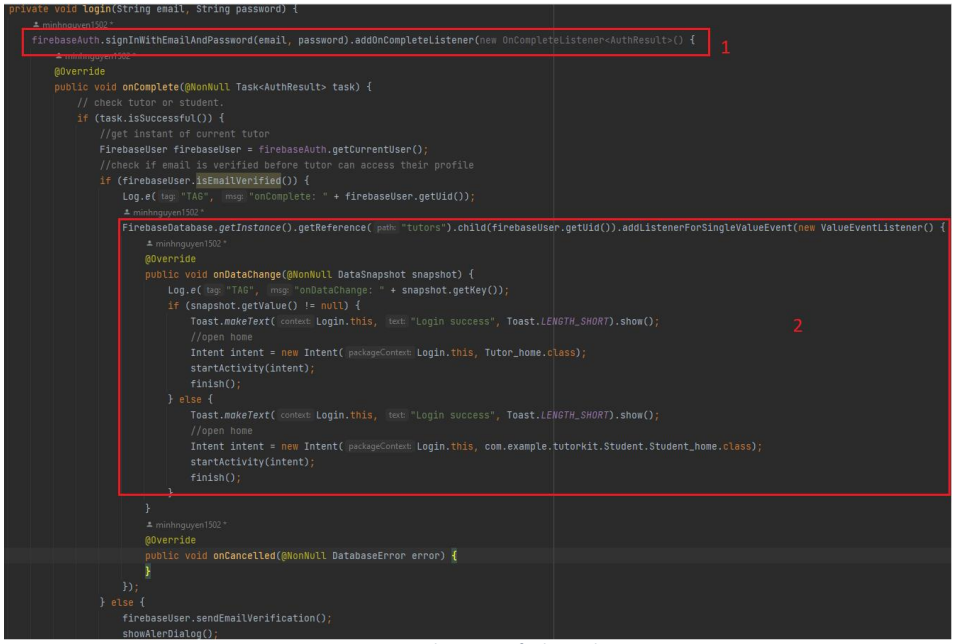
In the project, I initialize FirebaseAuth.



```
private void registerTutor(String txt_name, String txt_email,
                           String txt_dob, String txt_gender,
                           String txt_phone, String txt_address,
                           String txt_subject, String txt_intro,
                           String txt_password, Uri img) {
        FirebaseAuth firebaseAuth = FirebaseAuth.getInstance();

      final  StorageReference imgReference  =  storageReference.child(System.currentTimeMillis()+"."+getFilesExtension(img));
        // create profile
        firebaseAuth.createUserWithEmailAndPassword(txt_email, txt_password).addOnCompleteListener( activity: Tutor_register.this,
                 minhnguyen1502 *
                new OnCompleteListener<AuthResult>() {
                 minhnguyen1502
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    if (task.isSuccessful()) {

                        FirebaseUser firebaseUser = firebaseAuth.getCurrentUser();
```

Figure 40: register using firebaseAuth

To register a new user with information of user and include: “email and password”



```
private void login(String email, String password) {
    minhnguyen1502 *
    firebaseAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {   1
        minhnguyen1502 *
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            // check tutor or student.
            if (task.isSuccessful()) {
                //get instant of current tutor
                FirebaseUser firebaseUser = firebaseAuth.getCurrentUser();
                //check if email is verified before tutor can access their profile
                if (firebaseUser.isEmailVerified()) {
                    Log.e( tag: "TAG",  msg: "onComplete: " + firebaseUser.getUid());
                    minhnguyen1502 *
                    FirebaseDatabase.getInstance().getReference( path: "tutors").child(firebaseUser.getUid()).addListenerForSingleValueEvent(new ValueEventListener() {
                        minhnguyen1502 *
                        @Override
                        public void onDataChange(@NonNull DataSnapshot snapshot) {
                            Log.e( tag: "TAG",  msg: "onDataChange: " + snapshot.getKey());
                            if (snapshot.getValue() != null) {
                                Toast.makeText( context: Login.this,  text: "Login success", Toast.LENGTH_SHORT).show();
                                //open home
                                Intent intent = new Intent( packageContext: Login.this, Tutor_home.class);
                                startActivity(intent);
                                finish();
                            } else {
                                Toast.makeText( context: Login.this,  text: "Login success", Toast.LENGTH_SHORT).show();
                                //open home
                                Intent intent = new Intent( packageContext: Login.this, com.example.tutorkit.Student.Student_home.class);
                                startActivity(intent);
                                finish();
                            }
                        }
                        minhnguyen1502 *
                        @Override
                        public void onCancelled(@NonNull DatabaseError error) {
                        }
                    });
                } else {
                    firebaseUser.sendEmailVerification();
                    showAlerDialog();
```

Figure 41: login using firebseAuth

In box 1 I use firebaseAuth to log in. I use the code in box #2 to check if the currently logged in user is a tutor or a student by looking at their data in Firebase Realtime Database. Depending

on the presence of data, it redirects the user to the Tutor_home or Student_home activity and displays the congratulatory message "Login Successful".

```
firebaseAuth = FirebaseAuth.getInstance();
FirebaseUser firebaseUser = firebaseAuth.getCurrentUser();

String tutorID = firebaseUser.getUid();
```

Figure 42: get current user

I get the currently signed-in user.

```
private void createUser(FirebaseUser firebaseUser, String urlImage) {
    Tutor tutor = new Tutor(firebaseUser.getUid(), txt_name, txt_dob,
            txt_address, txt_phone, txt_gender, txt_subject, txt_intro,
            urlImage, new StatusAdd());
    minhnguyen1502
    referenceProfile.child(firebaseUser.getUid()).setValue(tutor).addOnCompleteListener(new OnCompleteListener<Void>() {
        minhnguyen1502
        @Override
        public void onComplete(@NonNull Task<Void> task) {

            if (task.isSuccessful()) {
                // sent verification to mail
                firebaseUser.sendEmailVerification();
                Toast.makeText( context: Tutor_register.this,  text: "Register success. please verify email", Toast.LENGTH_SHORT).show();

            } else {
                Toast.makeText( context: Tutor_register.this,  text: "Register Failed. ", Toast.LENGTH_SHORT).show();

            }
        }
    });
```

Figure 43: code VerifyEmail

In box I send a verification email to the user.

```
private void forgotPassword(String email) {
    firebaseAuth  = FirebaseAuth.getInstance();
    minhnguyen1502
    firebaseAuth.sendPasswordResetEmail(email).addOnCompleteListener(new OnCompleteListener<Void>() {
        minhnguyen1502
        @Override
        public void onComplete(@NonNull Task<Void> task) {
            if (task.isSuccessful()){
                Toast.makeText( context: ForgotPassword.this,  text: "Check inbbox for password reset link", Toast.LENGTH_SHORT).show();
                Intent i = new Intent( packageContext: ForgotPassword.this, Login.class);
                i.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_CLEAR_TASK
                        | Intent.FLAG_ACTIVITY_NEW_TASK);
                startActivity(i);
                finish();
            }else {
                try {
                    throw task.getException();

                }catch (FirebaseAuthInvalidUserException e){
                    edt_email.setError("not Exist");
                }catch (Exception e){
                    Log.e( tag: "Forgot password activity", e.getMessage());
                    Toast.makeText( context: ForgotPassword.this, e.getMessage(), Toast.LENGTH_SHORT).show();

                }
            }
        }
    });
```

Figure 44: code of reset password

Send a password reset email to a user.

6.3.3 Integrate card payments in Android apps

```
//lib
implementation 'de.hdodenhof:circleimageview:3.1.0'
implementation 'com.intuit.sdp:sdp-android:1.1.0'
implementation 'com.github.JessYanCoding:AndroidAutoSize:v1.2.1'
implementation 'androidx.appcompat:appcompat:1.2.0'
implementation 'com.squareup.picasso:picasso:2.5.2'
implementation 'com.lionscribe.open.libphonenumber:libphonenumber:8.12.18.1'
implementation 'com.github.bumptech.glide:glide:4.16.0'
implementation('com.paypal.checkout:android-sdk:1.1.0')
```

Figure 45: import SDK of paypal

Add the PayPal Android SDK dependency to your build.gradle

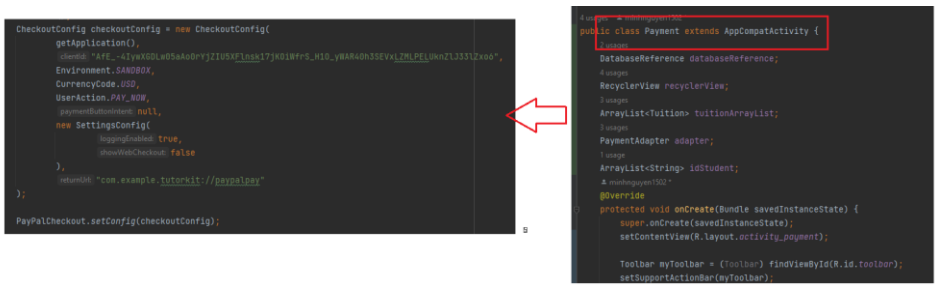


Figure 46: config

In class Payment, initialize the PayPal Service.

```
<com.paypal.checkout.paymentbutton.PaymentButtonContainer
    android:id="@+id/payment_button_container"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true"
    app:paypal_button_color="blue"
    app:paypal_button_enabled="true"
    app:paypal_button_label="pay"
    app:paypal_button_shape="rectangle"
    app:paypal_button_size="large" />
```

Figure 47: button PayPal

Create a button to initiate the PayPal payment

```
paymentButtonContainer.setup(
        new CreateOrder() {
            @Override
            public void create(@NotNull CreateOrderActions createOrderActions) {
                Log.d(TAG,  msg: "create: ");D
                ArrayList<PurchaseUnit> purchaseUnits = new ArrayList<>();
                purchaseUnits.add(
                        new PurchaseUnit.Builder()
                                .amount(
                                        new Amount.Builder()
                                                .currencyCode(CurrencyCode.USD)
                                                .value(String.valueOf(total))
                                                  .value("10.00")
                                                .build()
                                )
                                .build()
                );
                OrderRequest order = new OrderRequest(
                        OrderIntent.CAPTURE,
                        new AppContext.Builder()
                                .userAction(UserAction.PAY_NOW)
                                .build(),
                        purchaseUnits
                );
                createOrderActions.create(order, (CreateOrderActions.OnOrderCreated) null);
            }
        },
        new OnApprove() {
            @Override
            public void onApprove(@NotNull Approval approval) {
                approval.getOrderActions().capture(new OnCaptureComplete() {
                    1 usage
                    @Override
                    public void onCaptureComplete(@NotNull CaptureOrderResult result) {
                        Log.d(TAG, String.format("CaptureOrderResult: %s", result));
                        Toast.makeText( context: PayPalPaymentActivity.this,  text: "Successful", Toast.LENGTH_SHORT).show();
                        updateStatusInFirebase();

                    }
                });
            }
```

Figure 48: handle the payment process

handle the payment process

```
private void updateStatusInFirebase() {
    // Assuming you have an ID for the tuition or a way to uniquely identify the tuition
    // You can pass this ID from the previous activity or get it from somewhere else
    String tuitionId = getIntent().getStringExtra( name: "tuitionId"); // You need to pass this from the previous activity

    if (tuitionId != null) {
        databaseReference.child( pathString: "tuition").child(tuitionId).child( pathString: "status").setValue(true)
                .addOnSuccessListener(new OnSuccessListener<Void>() {
                    @Override
                    public void onSuccess(Void unused) {
                        Log.d(TAG,  msg: "Status updated successfully");
                    }
                })
                .addOnFailureListener(new OnFailureListener() {
                    @Override
                    public void onFailure(@NonNull Exception e) {
                        Log.e(TAG,  msg: "Failed to update status", e);
                    }
                });
    } else {
        Log.e(TAG,  msg: "Tuition ID is null");
    }
}
```

Figure 49: update status of tuition

Update the status of the tuition invoice.

6.3.4 Explain the libraries used in the project

```
45      implementation 'de.hdodenhof:circleimageview:3.1.0'
46      implementation 'com.intuit.sdp:sdp-android:1.1.0'
47      implementation 'com.github.JessYanCoding:AndroidAutoSize:v1.2.1'
48      implementation 'com.lionscribe.open.libphonenumber:libphonenumber:8.12.18.1'
49      implementation 'com.github.bumptech.glide:glide:4.16.0'
50      implementation( com.paypal.checkout:android-sdk:1.1.0')
51
```

Figure 50: libs

6.3.4.1 CircleImageView:

CircleImageView is a custom ImageView that creates a circular image from any Drawable.

It's useful for displaying circular profile pictures or images in a circular shape.

```
<de.hdodenhof.circleimageview.CircleImageView
    android:id="@+id/avatar"
    android:layout_width="@dimen/_100sdp"
    android:layout_height="@dimen/_100sdp"
    android:layout_marginTop="@dimen/_20sdp"
    android:layout_marginStart="@dimen/_100sdp"
    android:layout_marginEnd="@dimen/_100sdp"
    android:src="@drawable/avata"
    app:layout_constraintBottom_toTopOf="@+id/scrollView2"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"/>
```

Figure 51: using CircleImageView in project

6.3.4.2 SDP (Scalable DP)

SDP stands for Scalable DP. It's a scalable size unit similar to dp but scaled with the screen density. It helps to create responsive UIs by scaling the size of UI elements based on screen density.

```
<de.hdodenhof.circleimageview.CircleImageView
    android:id="@+id/avatar"
    android:layout_width="@dimen/_100sdp"
    android:layout_height="@dimen/_100sdp"
    android:layout_marginTop="@dimen/_20sdp"
    android:layout_marginStart="@dimen/_100sdp"
    android:layout_marginEnd="@dimen/_100sdp"
    android:src="@drawable/avata"
    app:layout_constraintBottom_toTopOf="@+id/scrollView2"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"/>
```

Figure 52:using SPD in project

6.3.4.3 AndroidAutoSize

AndroidAutoSize is a dynamic font size solution for Android, which matches different screen sizes and resolutions. It automatically adjusts the size of the text and other UI elements to fit various screen sizes, improving the app's adaptability.

6.3.4.4 libphonenumber

libphonenumber is a library that provides utilities for parsing, formatting, and validating phone numbers. It's useful for working with phone numbers, validating them, formatting them according to international standards, and more.

```
boolean isValid = false;
try {
    PhoneNumberUtil phoneUtil = PhoneNumberUtil.getInstance( context: Student_register.this);
    try {
        swissNumberProto = phoneUtil.parse(txt_phone, Locale.getDefault().getCountry());
    } catch (NumberParseException e) {
        System.err.println("NumberParseException was thrown: " + e);
    }
    isValid = phoneUtil.isValidNumber(swissNumberProto); // returns true

} catch (Exception e) {
    e.printStackTrace();
}
```

Figure 53:using libphonenumber in project

6.3.4.5 Glide

Glide is an image loading and caching library for Android.

```
Glide
        .with( activity: Student_profile.this)
        .load(student.getImg())
        .centerCrop()
        .into(avatar);
```

Figure 54: using glide in project

6.3.4.6 PayPal Checkout SDK

PayPal Checkout SDK provides tools and components to integrate PayPal payments into Android apps.

6.4 Development plan

Phase 1: Requirements Gathering and Analysis

During this initial phase, we engage with stakeholders to understand their needs and expectations for the project. This involves conducting interviews, surveys, and meetings to capture detailed requirements. Concurrently, we conduct a comprehensive literature review to understand the domain-specific knowledge and the latest technologies and methodologies relevant to the project. This research culminates in a Literature Review document that serves as a foundation for the project. We identify key project attributes such as objectives, aims, and scope to establish clear project boundaries. Based on the gathered requirements and findings from the literature review, we draft the System Requirement Specification (SRS) document. Before proceeding further, we seek user acceptance to ensure alignment with their expectations.

Phase 2: System Design and Architecture

In the second phase, we focus on designing the system architecture that outlines the overall structure and components of the software. We create a Use Case Diagram to visualize interactions between users and the system, aiding in understanding user interactions and system responses. Database design is also crucial at this stage; we normalize and design the database schema to ensure efficient data storage and retrieval. Additionally, we design user interfaces (UI) that are intuitive and user-friendly, enhancing the overall user experience. As with the previous phase, we obtain user acceptance to validate the design documents, ensuring they meet user requirements and expectations.

Phase 3: Implementation

Implementation involves translating the design and requirements into actual software. Our development team works diligently to code the software based on the approved design and requirements documents. Throughout the development process, we perform unit tests to identify and rectify issues at an early stage. We also integrate third-party services or APIs as required to enhance the functionality of the software. By the end of this phase, we achieve the milestone of having a completely developed software ready for testing and validation.

Phase 4: Test and Fix Bugs

Testing is a critical phase where we ensure the software meets quality standards and functions as intended. We begin by writing a detailed test plan that includes test scenarios, test cases, and test logs to guide our testing efforts. We execute the tests based on the test plan,

simulating various user interactions and scenarios to identify any issues or bugs. We actively involve stakeholders in the testing process to gather feedback and ensure the software aligns with their expectations. As bugs are identified, we report them to the development team, prioritizing fixes based on their severity and impact on the software.

Phase 5: Create Documents

The final phase focuses on documenting the entire project, consolidating information from all previous phases. We gather documentation including the Literature Review, SRS, design documents, test plans, and test logs. A User Manual is created to guide users on how to use the software effectively, providing step-by-step instructions and troubleshooting tips. Before finalizing the documents, we obtain user acceptance to ensure accuracy and completeness. Once approved, these documents serve as valuable resources for stakeholders and future project teams.

7 Evaluation

7.1 Result:

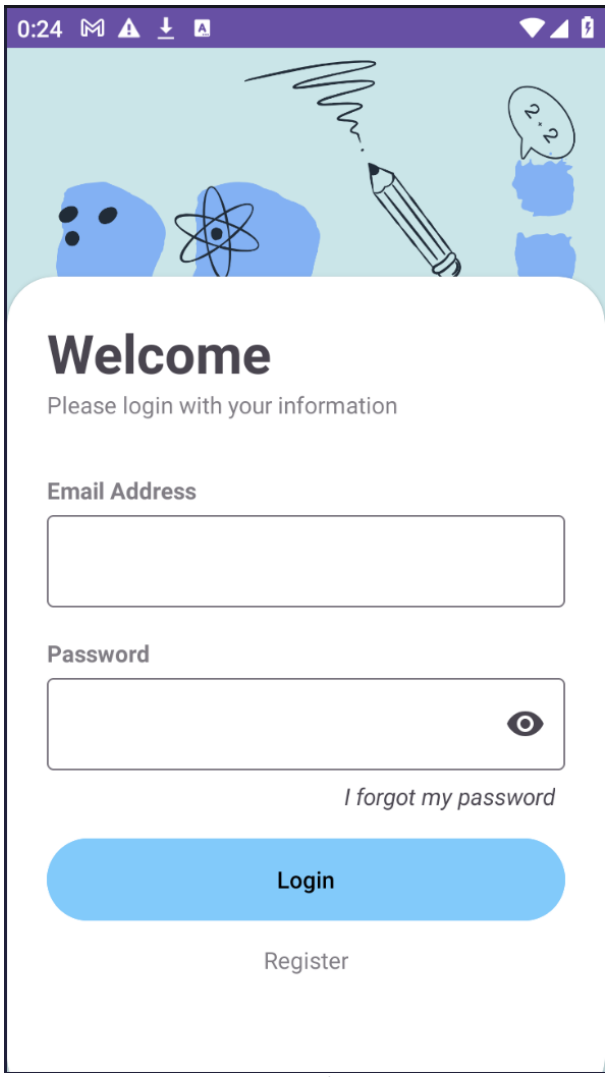


Figure 55: login

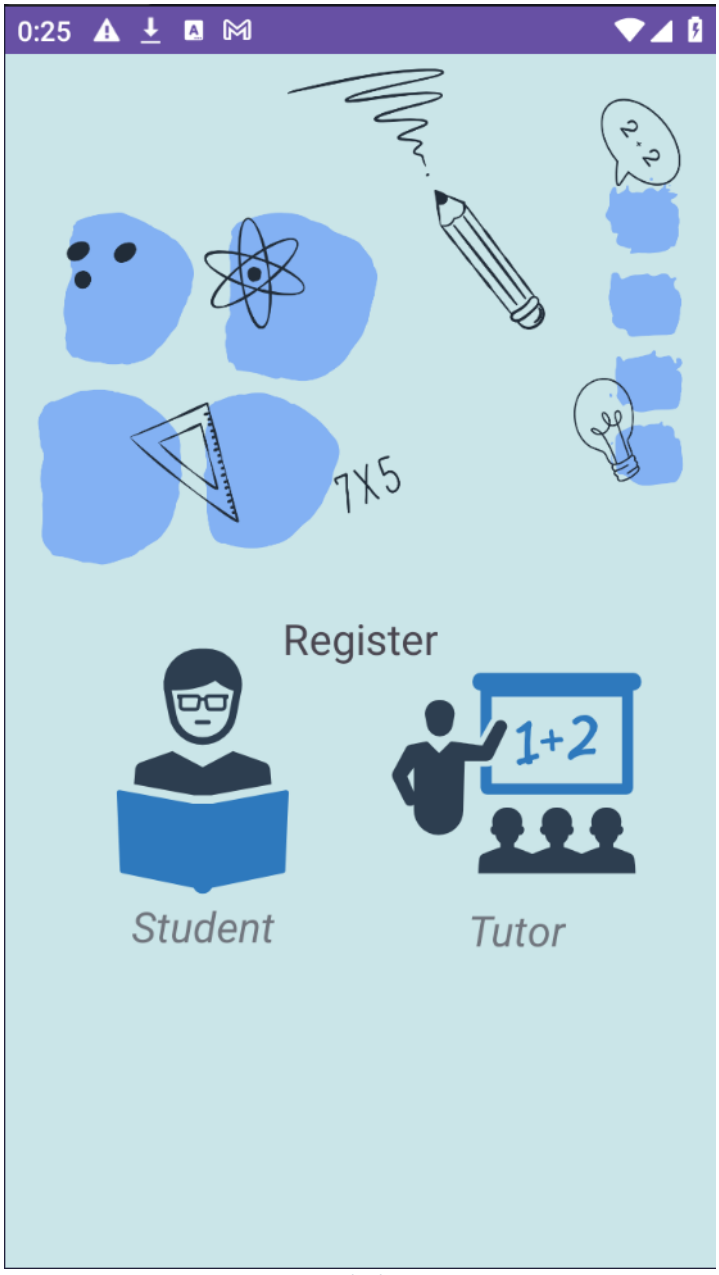


Figure 56: pick role to register

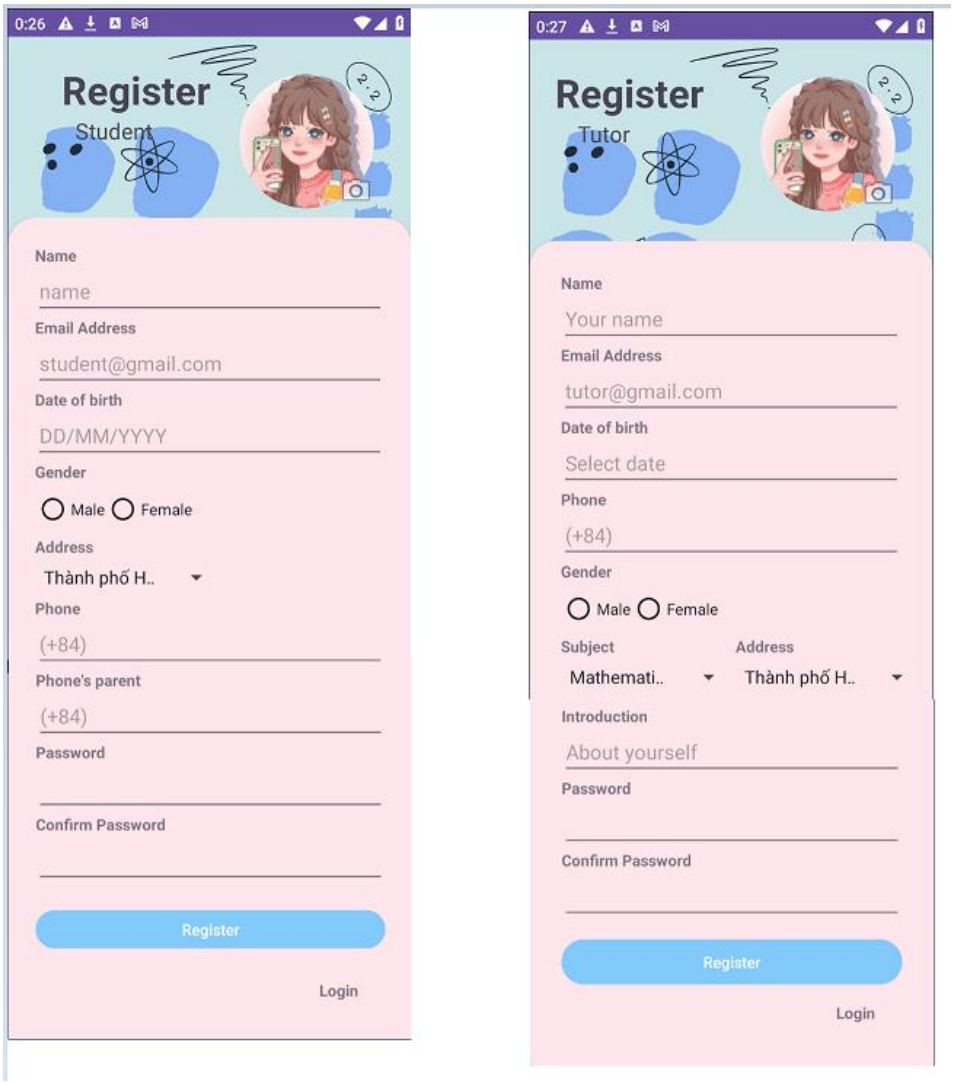


Figure 57: register 2 roles

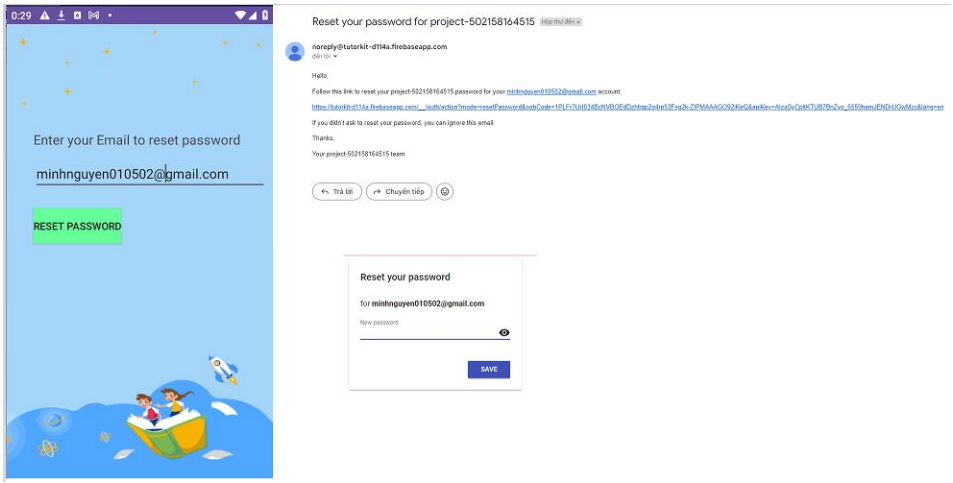
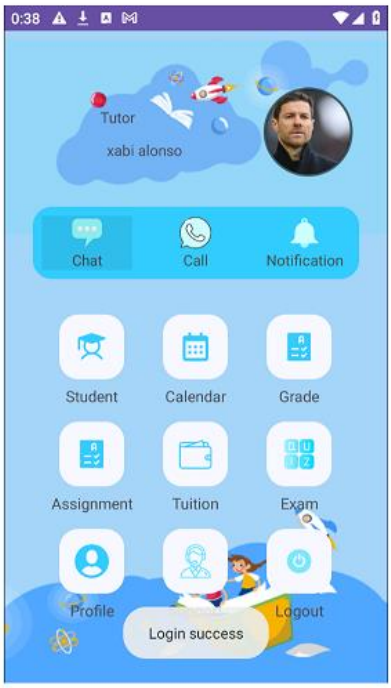


Figure 58: forgot password

Tutor



Student

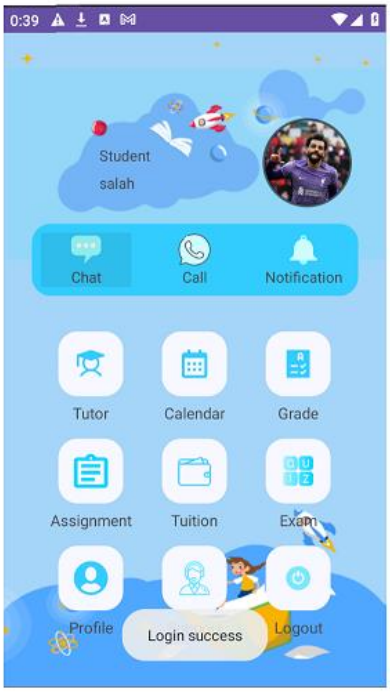
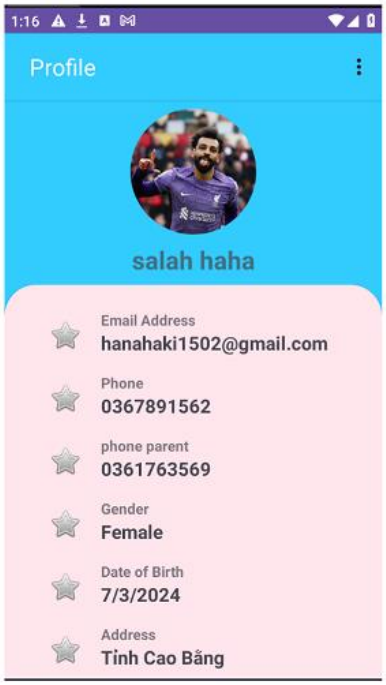


Figure 59: home page

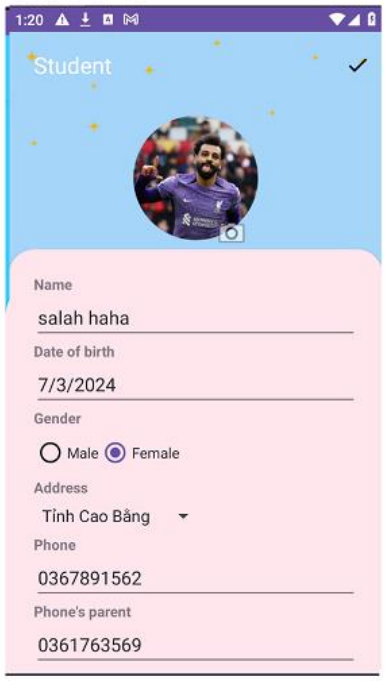
Student



Tutor

Figure 60: profile

Student



Tutor

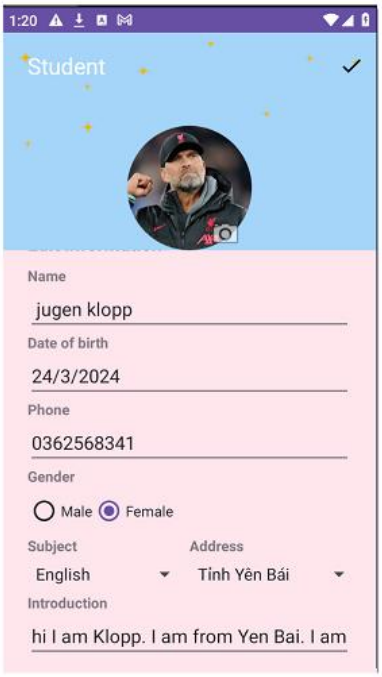


Figure 61: edit information

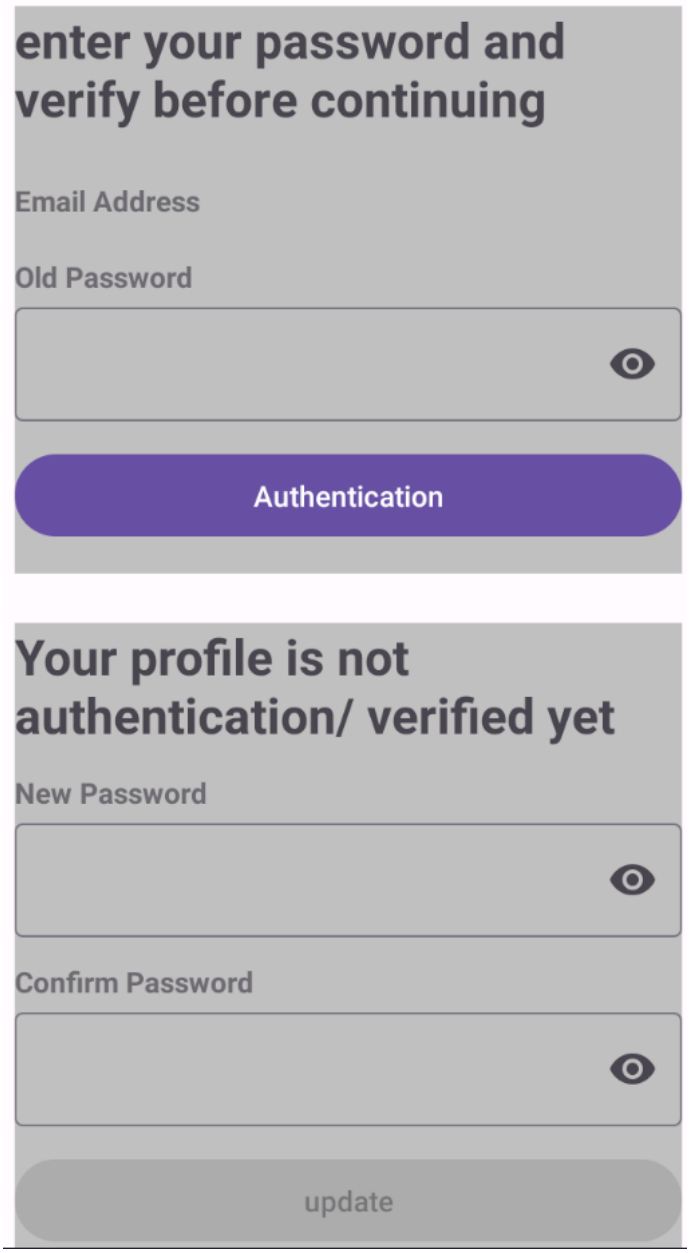


Figure 62: update password

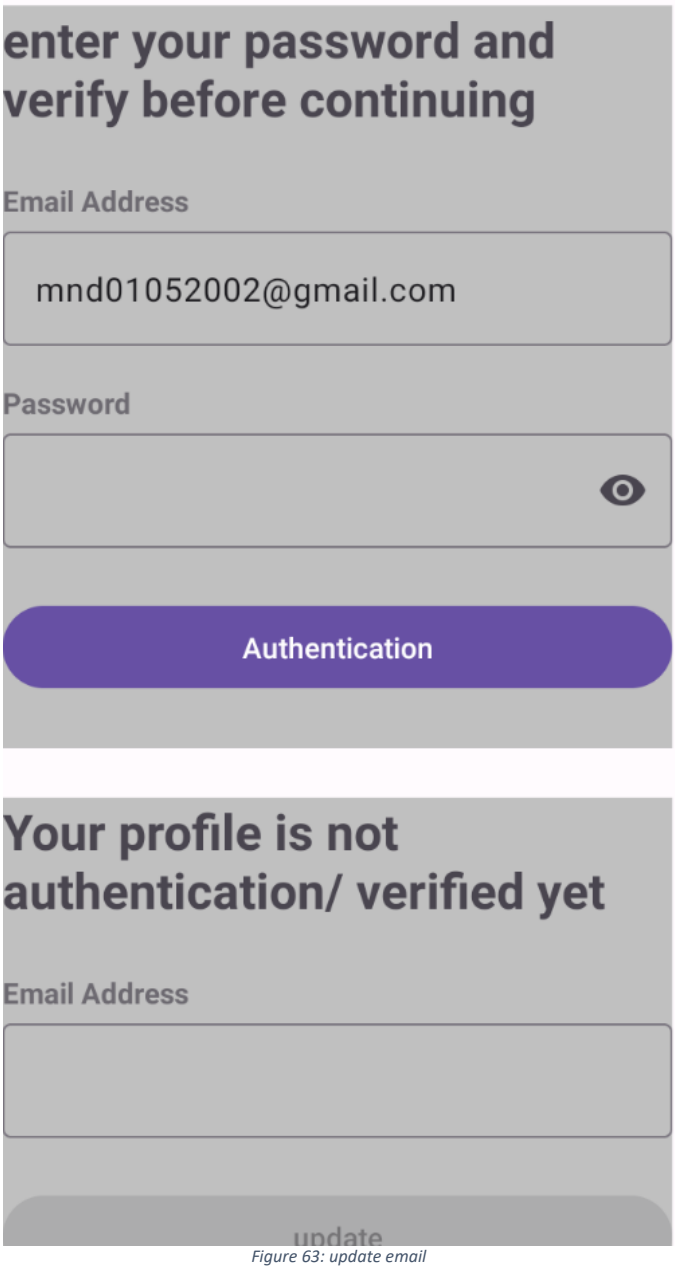


Figure 63: update email

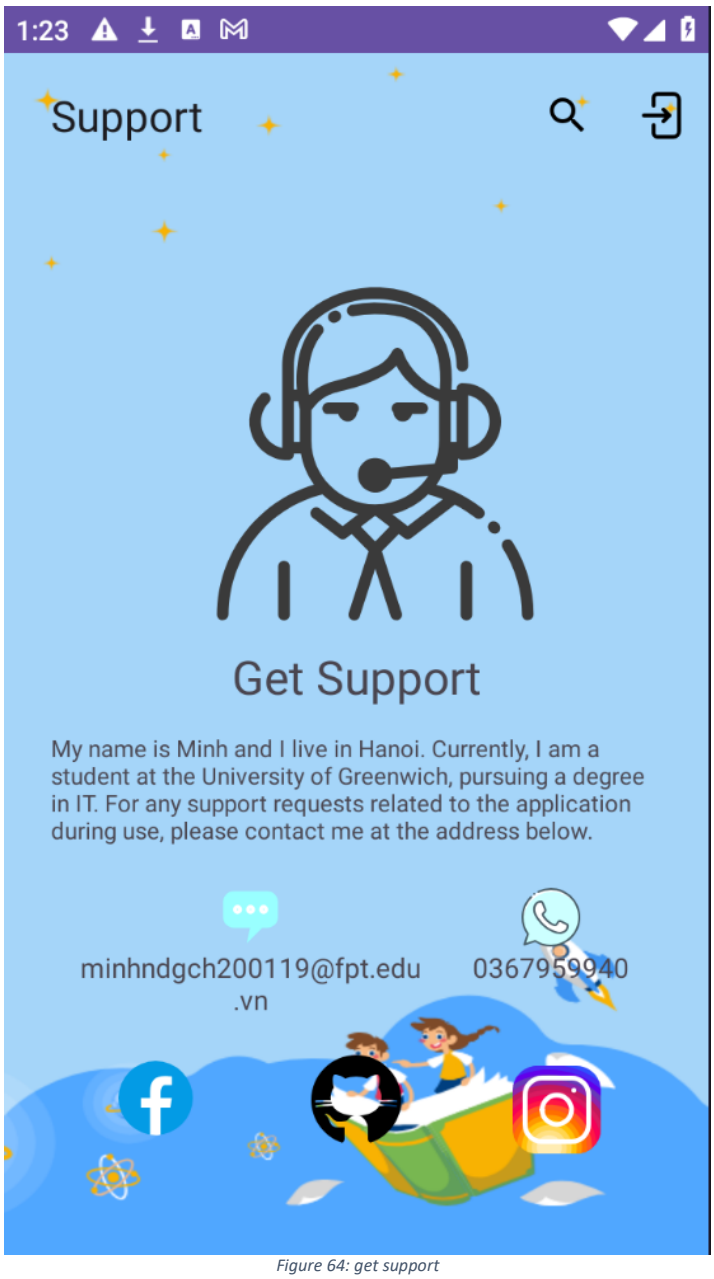
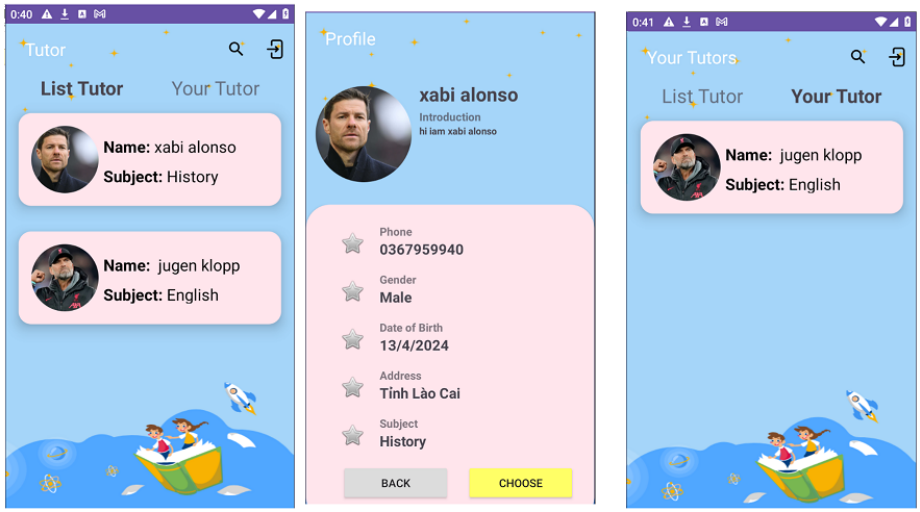


Figure 64: get support

Student



Tutor

Figure 65: approve student

Student

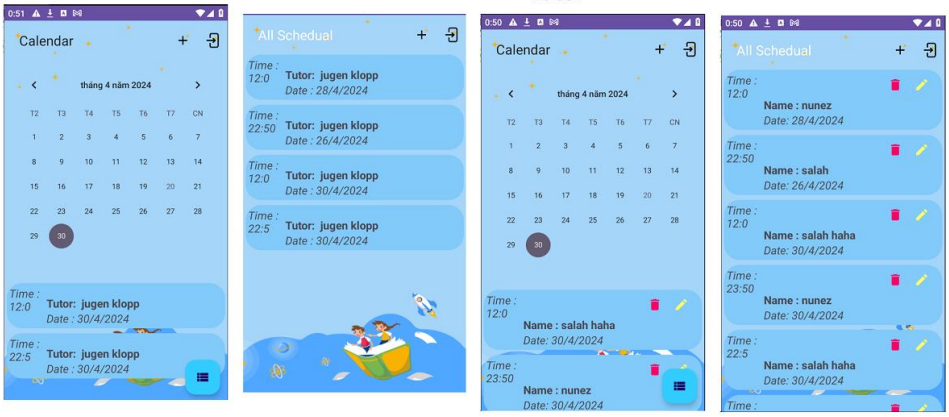
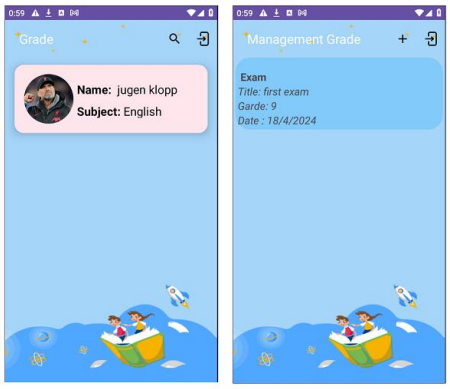


Figure 66: schedule

Student



Tutor

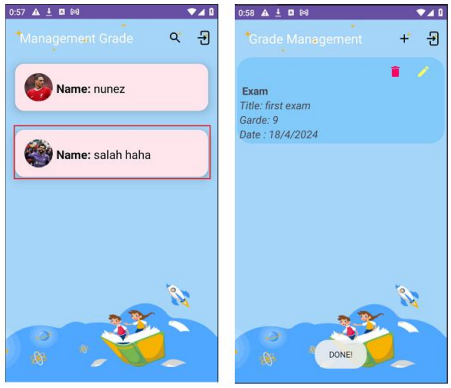


Figure 67: grade page

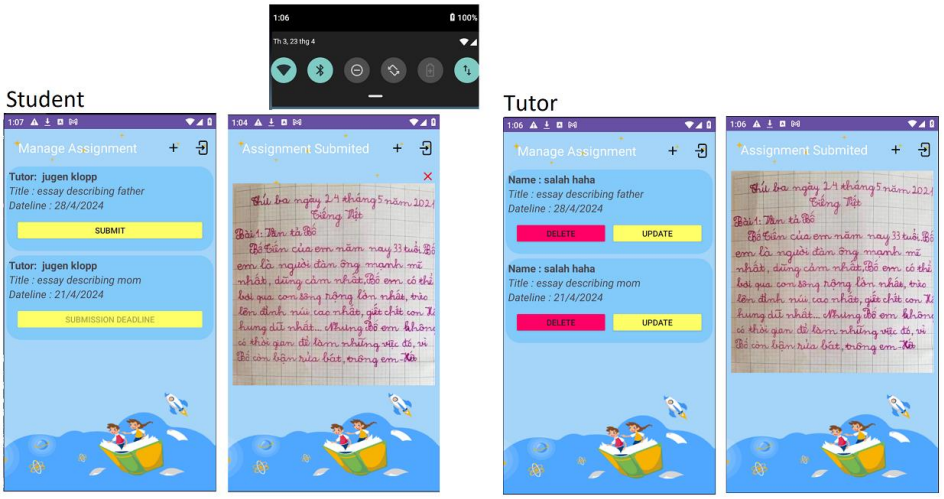


Figure 68: assignment

Student

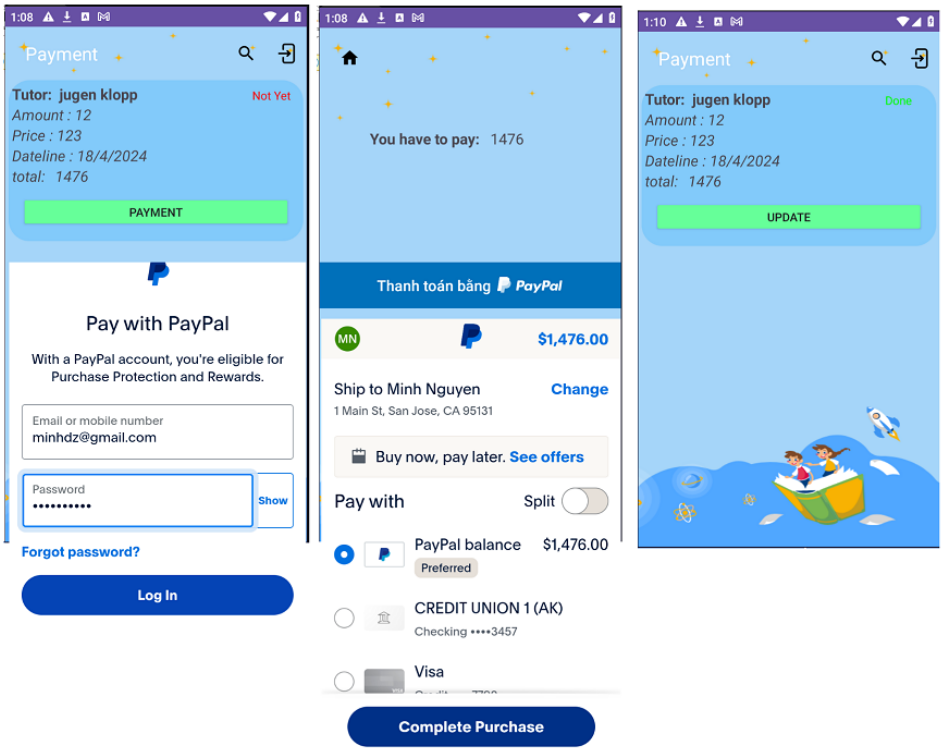


Figure 69: student payment

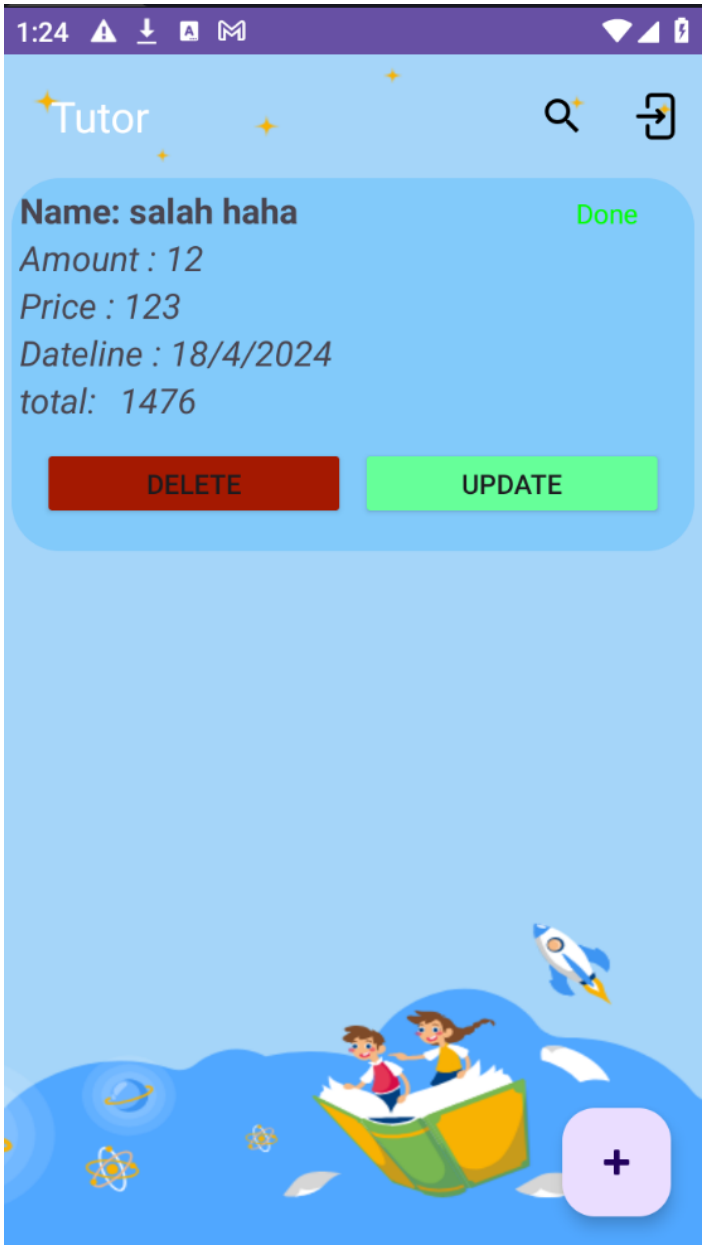


Figure 70: list Tuition of tutor

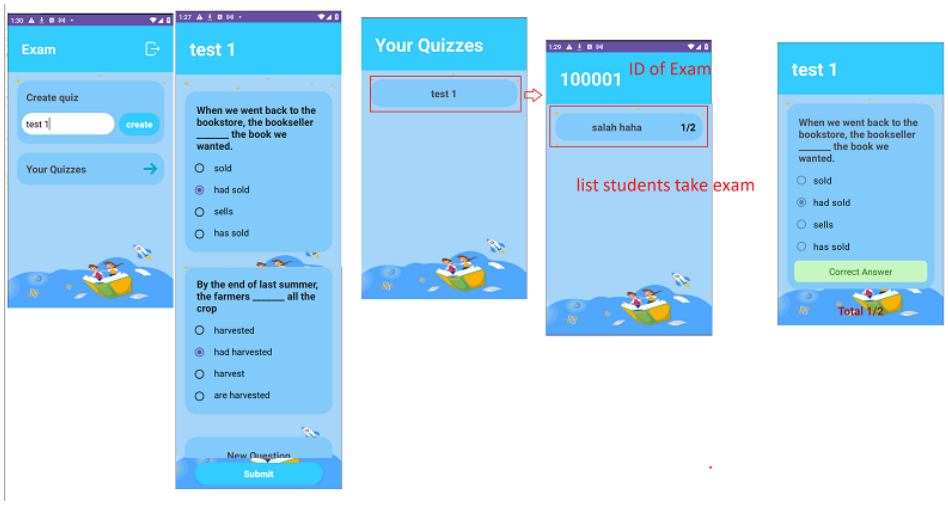


Figure 71: create an exam

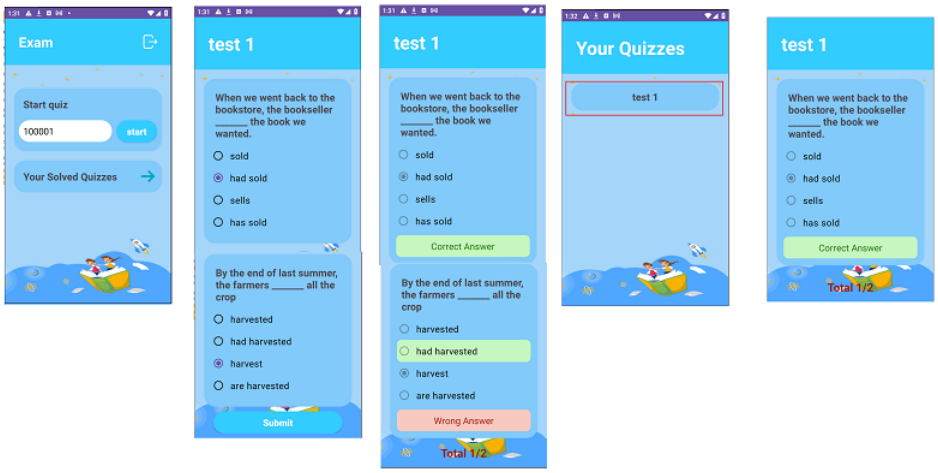


Figure 72: take an exam

Tutor

Student

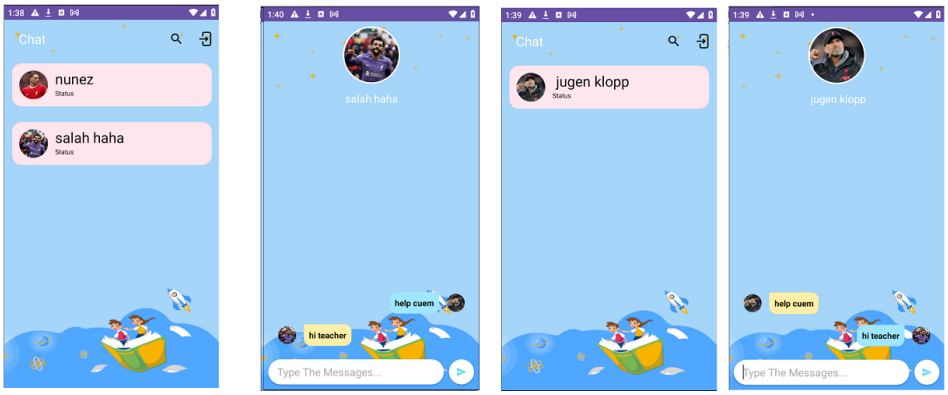


Figure 73: chat

Tutor

Student

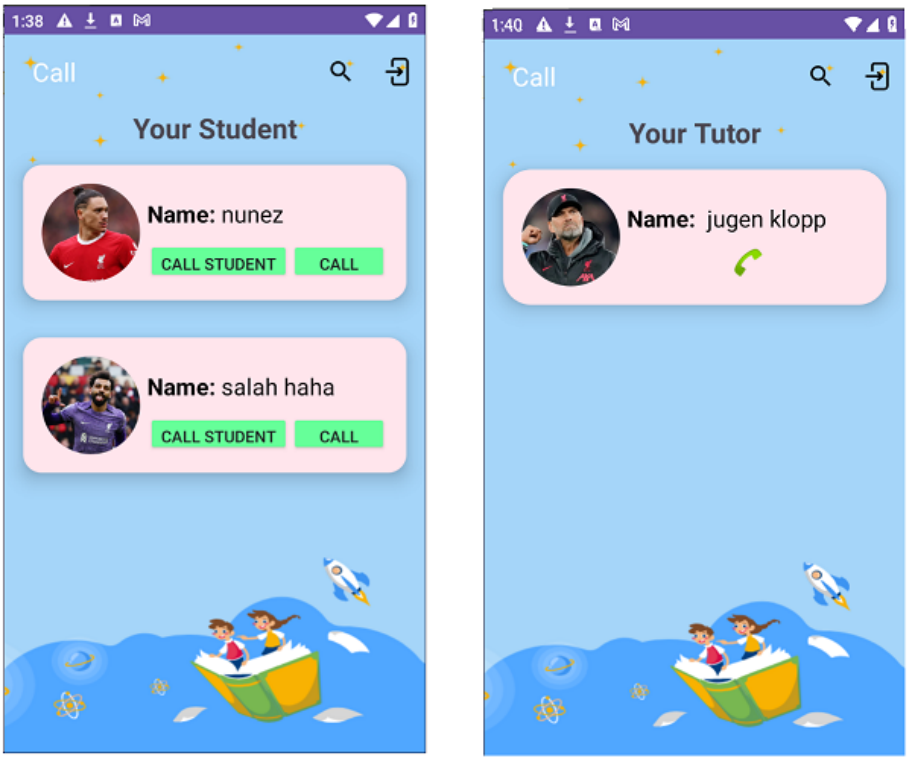
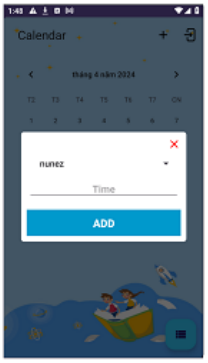


Figure 74: call

Schedule

Create



Update

Delete

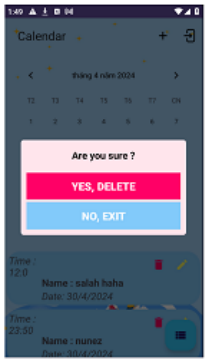
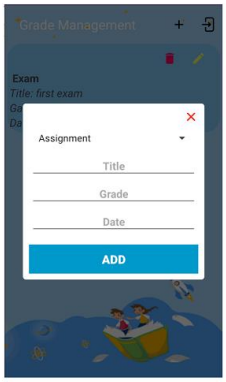


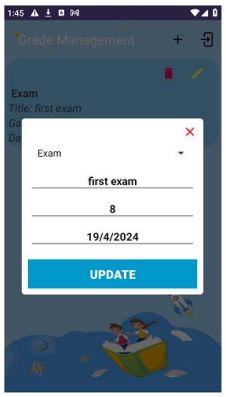
Figure 75: manage Schedule

Grade

Create



Update



Delete

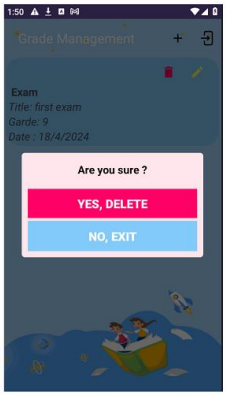
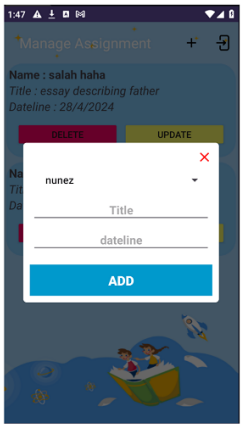
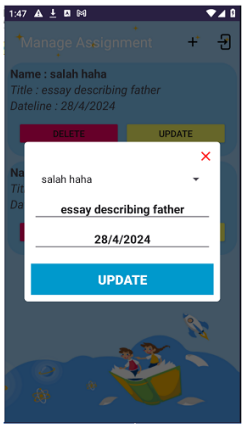


Figure 76: manage Grade

Create



Update



Delete

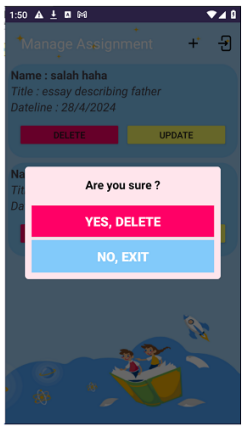
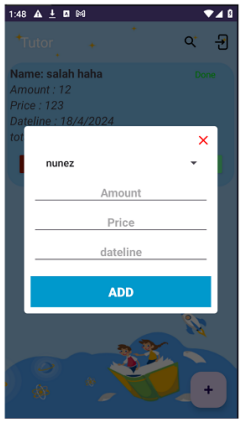
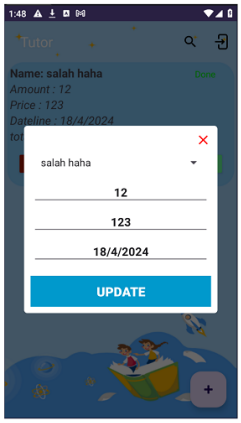


Figure 77: manage Assignment

Create



Update



Delete

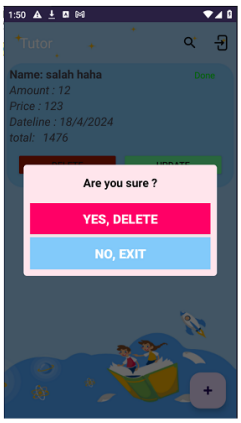


Figure 78: manage Tuition

7.2 Test:

7.2.1 Test plan

7.2.1.1 Introduction

TutorKit is a specialized educational platform catering to two primary roles: Tutors and Students. The primary objective of this testing is to identify and rectify any issues, bugs, or

inconsistencies that may affect the functionality, performance, or user experience of the TutorKit application. The testing process will encompass functional testing to validate the core functionalities, usability testing to assess the user-friendliness of the application, performance testing to evaluate the application's responsiveness and stability under load, and security testing to ensure the confidentiality and integrity of user data. This test plan is flexible and subject to modifications based on the findings during the testing process, changes in project scope, or updates to the application requirements. The testing team will collaborate closely with the development team to communicate any changes or updates to the test plan promptly and ensure that the testing process aligns with the project's goals and objectives.

7.2.1.2 Test strategy

7.2.1.2.1 Objective

The objective of the testing for TutorKit is to ensure that the application's functionality aligns with the user requirements and to identify and rectify any potential issues or bugs that could impact the system's operation. The testing process will focus on validating the test cases, executing and validating the test scenarios, addressing and retesting high and medium severity defects based on predefined criteria, and prioritizing low-severity bugs for future maintenance.

Output of the Testing Phase:

- Production-Ready Application: A thoroughly tested TutorKit application that is ready for deployment in a production environment, ensuring reliability and performance.
- Comprehensive Testing Documentation: A collection of documents, including testing scripts and test cases, which can be reused in future User Acceptance Testing (UAT) or for reference during maintenance and updates.

7.2.1.2.2 Scope

This test plan will encompass the testing of the TutorKit software application. The scope of the testing will be comprehensive, addressing various aspects of the application to ensure its functionality, usability, performance, and security.

Scope of Testing:

- Functional Testing: This phase will focus on testing all the functional requirements of TutorKit, including data input, output, and data processing.

- Usability Testing: This section will concentrate on evaluating the user interface (UI) and user experience (UX) of TutorKit.
- Performance Testing: This phase will cover the testing of TutorKit's performance metrics, including response time, processing speed, and memory usage.
- Security Testing: This section will focus on testing the security features of TutorKit to safeguard user data and ensure the confidentiality and integrity of information.

7.2.1.2.3 Test environment

Hardware	• Window 10 or more. • Intel(R) Core(TM) i3-10105F CPU @ 3.70GHz 3.70 GHz • 8.00 GB RAM
Software	• Android Studio • Java

Table 9: test environment

7.2.2 Testing implementation

Function	ID	Description	Data	Expected result	Actual result	Status
Register	R-001	Not filling in the input and clicking the submit button.		Warning for filling in missing information.	Warning for filling in missing information.	Pass
	R-002	Verify the phone number according to the device's language (for example, if the phone is set to Vietnam, you must fill in the phone number according to Vietnam such as 036 XXX XXXX, 098 XXX XXXX,....)	0367959940	Success (qualification)	Don't have notification	Pass
			0147859632	Notification (Phone no. is not valid) and Re-enter	Notification (Phone no. is not valid) and Re-enter	Pass
	R-003	Check phone number have 10 numbers	0367959940	Success (qualification)	Don't have notification	Pass
			0367954	Notification (Phone no. is not valid) and Re-enter	Notification (Phone no. is not valid) and Re-enter	Pass

	R-005	Check email already exists	Enter : minhnguyen010502@gmail.com Email exists: minhnguyen010502@gmail.com	Notify: enter another email	Notify: enter another email	Pass
	R-004	The password field must have a minimum of 6	uiojkl	Success (qualification)	Don't have notification	Pass
			uio	Notification (password is too weak) and Re-enter	Notification (password is too weak) and Re-enter	Pass
	R-005	Verify user-added passwords must be protected, encrypted, and displayed with an asterisk (******).	uiojkl	Enter: uiojkl Display : ******	Enter: uiojkl Display : ******	Pass
	R-006	Verify if authentication is added for the password and confirm if the passwords are the same.	Password: uiojkl Cf-Password: uiojkl	Success (qualification)	Don't have notification	Pass
			Password: uiojklbnm Cf-Password: uiojkl	Notification (password confirm must be the same the password)	Notification (password confirm must be the same the password)	Pass
			Password: uiojkl Cf-Password: uiojklbnm			
	R-007	Verify the email verification link has been sent to the user's email address successfully.		Send a verification link to the email the person has registered with.	Send a verification link to the email the person has registered with.	Pass
Forgot password	FP-001	Send link to email was enteted	minhnguyen010502@gmail.com	Send link reset password to email	Send link reset password to email	pass
	FP-002	Verify if authentication is added for the password and confirm if the passwords are the same.	Password: uiojkl Cf-Password: uiojkl	Success (qualification)	Don't have notification	Pass
			Password: uiojklbnm Cf-Password: uiojkl	Notification (password confirm must be the same the password)	Notification (password confirm must be the same the password)	Pass
			Password: uiojkl Cf-Password: uiojklbnm			
	FP-003	Login again with new password	uiojkl	Success (qualification)	Don't have notification	Pass
Login	LI-001		True:	Login success	Login success	Pass

		Verify that the user can log in by entering credentials and clicking the Log In button.	Email: minhnguyen010502@gmail.com Password: uiojkl			
			False: Email: asdfasf@gmail.com Password: uiojkl	Don't login	Don't login	Pass
	LI-002	Check verifies user's email	Email was verified	Login success	Login success	Pass
			Email wasn't verified	Open dialog open email	Open dialog open email	Pass
	LI-003	Check device have app email	yes	Open app email	Open app email	Pass
			no	Notify (don't have app)	Notify (don't have app)	Pass
	LI-004	Verify that the entered password is in encrypted form (******).	uiojkl	Display ******	Display ******	Pass
	LI-005	Verify that the user can see the password by clicking the eye icon.	Display ******	Display: uiojkl	Display: uiojkl	Pass
	LI-006	Verify the error message that displays after leaving the passcode or password field blank.		Notify enter email and password	Notify enter email and password	Pass
	LI-007	Verify authorization when logging in by role.	Email Student	Open page Student home	Open page Student home	Pass
			Email Tutor	Open page Tutor home	Open page Tutor home	Pass
Logout	Lo-001	Logout		Return login page	Return login page	Pass
Function account user	FA-001	View Profile	Data of account	Display information	Display information	Pass
	FA-002	Update profile	Data of account	View data before update	View data before update	Pass
	FA-003	Not filling in the input and clicking the submit button.		Warning for filling in missing information.	Warning for filling in missing information.	Pass

	FA-004	Verify the phone number according to the device's language (for example, if the phone is set to Vietnam, you must fill in the phone number according to Vietnam such as 036 XXX XXXX, 098 XXX XXXX, ...)	0367959940	Success (qualification)	Don't have notification	Pass
			0147859632	Notification (Phone no. is not valid) and Re-enter	Notification (Phone no. is not valid) and Re-enter	Pass
	FA-005	Check phone number have 10 numbers	0367959940	Success (qualification)	Don't have notification	Pass
			0367954	Notification (Phone no. is not valid) and Re-enter	Notification (Phone no. is not valid) and Re-enter	Pass
	FA-006	Update Email	Correct password	Button Authentication cannot be clicked and the new email input box and button Update are opened.	Button Authentication cannot be clicked and the new email input box and button Update are opened.	Pass
			Incorrect password	Error and re-enter	Error and re-enter	Pass
	FA-007	Verify Email format.	asdasd	Notify enter valid email	Notify enter valid email	Pass
			minhnguyen010502@gmail.com	Back to page login and notify "Verify new email"	Back to page login and notify "Verify new email"	Pass
	FA-008	Login with new email	New Email was verified	Login success	Login success	Pass
			New Email wasn't verified	Open dialog open email	Open dialog open email	Pass
	FA-009	Update password	Incorrect old password	Error and re-enter	Error and re-enter	Pass
			Correct old password	Button Authentication cannot be clicked and the new	Button Authentication cannot be clicked and the new	Pass

				password and confirm password input box and button Update are opened.	password and confirm password input box and button Update are opened.	
			New password: uiojkl Old password : uiojkl	Notify: new password don't have to be the same old password	Notify: new password don't have to be the same old password	Pass
			New password:jklbnm Confirm password: jklbnmuio Or New password:jklbnmuio Confirm password: jklbnm	Notify: confirm password must be the same new password	Notify: confirm password must be the same new password	Pass
			New password:jklbnm Confirm password: jklbnm	Update success and back to page login	Update success and back to page login	Pass
	FA-010	The new password field must have a minimum of 6	New password:jkl	Notification (password is too weak) and Re-enter	Notification (password is too weak) and Re-enter	Pass
			New password:jklbnm	Update success and back to page login	Update success and back to page login	pass
	FA-011	Login with new password	jklbnm	Login success	Login success	Pass
			uiojkl	Login fail	Login fail	pass
Support	S-001	View developer information	Email address, phone, fb, git, insta	Display email address, phone, fb, git, insta	Display email address, phone, fb, git, insta	Pass
	S-002	Send email	Email	Clicking on the email address will open the email application and the incoming email address is the developer's email	Clicking on the email address will open the email application and the incoming email address is the developer's email	Pass
	S-003	Call to developer	Phone no.	Clicking on the phone number will open the	Clicking on the phone number will open the	Pass

				phone app and the developer's phone number will be displayed and just click call.	phone app and the developer's phone number will be displayed	
	S-004	Connect fb with developer	Link fb	Clicking on the fb icon will lead to the developer's fb	Clicking on the fb icon will lead to the developer's fb	Pass
	S-005	Connect insta with developer	Link insta	Clicking on the insta icon will lead to the developer's insta	Clicking on the insta icon will lead to the developer's insta	Pass
	S-006	Open source code of project	Link git	Clicking on the git icon will open git which saves the project's source code.	Clicking on the git icon will open git which saves the project's source code.	Pass
Function Student	FS-001	View list Tutors	All tutors	All tutors not approve in database display	Button choose the tutor was hired	Pass
	FS-002	Choose tutor		Information's student display in page list Student in account Tutor who that student clicked choose	Information's student display in page list Student in account Tutor who that student clicked choose	Pass
	FS-003	View list student's tutors	List tutors approved	The list of approved tutors is displayed on the "Your tutor" page.	The list of approved tutors is displayed on the "Your tutor" page.	Pass
	FS-004	View student's grade (click to tutor created grade)	Grade (grade was created by tutor's that student)	List student's grades displayed	List student's grades displayed	Pass
	FS-005	View list Assignments need to submit (all student's tutors)	List Assignments	List Assignments displayed	List Assignments displayed	Pass
	FS - 006	Verify that assignments past	Dateline: 4/4/2024 Today: 7/4/2024	Button Submit not working	Button Submit not working	Pass

		the deadline will not be submitted.				
	FS-007	Submit Assignment (click button plus to choose image)	Image's assignment	Display list image's assignment when click button add in dialog	Display list image's assignment	Pass
	FS-008	Tuition invoices need to be paid.	Tuition invoices	Displays the student's tuition invoice that needs to be paid.	Displays the student's tuition invoice that needs to be paid.	Pass
	FS-009	Payment		Payment with PayPal	Payment with PayPal	Pass
	FS-010	Take an exam	Id's exam	Displays the test with the student id just entered	displays the test with the student id just entered	Pass
			Id don't exist	Box input was cleared	Box input was cleared	Pass
	FS-011	View solved quizzes	The exam was solved	Display the exam was solved and result	Display the exam was solved and result	Pass
	FS-012	View schedule	The schedule	Displays the day of the month and a list of class schedules	Displays the day of the month and a list of class schedules	Pass
	FS-013	Highlight the day have Schedule		The day have schedule change color	The day have schedule doesn't change color	Fail
	FS-014	Chat with student's tutors		Tutor and student chat together	Tutor and student chat together	Pass
	FS-015	Call with student's tutors		Student call for tutor's the student by phone number	Student call for tutor's the student by phone number	Pass
Function tutor	FT-001	Approve student		Display list students need to approve	Display list students need to approve	Pass
	FT-002	View list students approved		Display list students approved in Your student page	Display list students approved in Your student page	Pass
	FT-002	Add grade's student.		Warning for filling in missing.	Warning for filling in missing.	Pass

		Not filling in the input and clicking the ADD button.				
	FT-003	Add grade's student. Garde >10 and <0	Grade: 12 Or grade: -1	Notify: grade must be less than 10 and greater than 0	Notify: grade must be less than 10 and greater than 0	Pass
	FT-004	Add grade's student.	Type: Assignment or exam (using spiner) Title: lesson 1 Grade: 7 Date: 4/4/2024 (picker)	Notify: done And display list grade's the student	Notify: done And display list grade's the student	Pass
	FT-005	Edit grade's the student. View before edit	Type: Assignment (using spiner) Title: lesson 1 Grade: 7 Date: 4/4/2024 (picker)	Display that grade's the student	Display that grade's the student	Pass
	FT-006	Edit grade's the student. Not filling in the input and clicking the ADD button.		Warning for filling in missing.	Warning for filling in missing.	Pass
	FT-007	Edit grade's the student. Garde >10 and <0	Grade: 12 Or grade: -1	Notify: grade must be less than 10 and greater than 0	Notify: grade must be less than 10 and greater than 0	Pass
	FT-008	Edit grade's the student.	New: Type: Exam (using spiner) Title: lesson 3 Grade: 8 Date: 8/4/2024 (picker)	Notify :update success and display: Type: Exam (using spiner) Title: lesson 3 Grade: 8 Date: 8/4/2024 (picker)	Notify :update success and display: Type: Exam (using spiner) Title: lesson 3 Grade: 8 Date: 8/4/2024 (picker)	Pass
	FT-009	Delete grade's student	Type: Exam (using spinner) Title: lesson 3 Grade: 8 Date: 8/4/2024 (picker)	That grade does not exist on screen nor on database	That grade does not exist on screen nor on database	Pass
	FT-010	Create an area to submit assignments.		Warning for filling in missing.	Warning for filling in missing.	Pass

		Not filling in the input and clicking the ADD button.				
	FT-011	Create an area to submit assignments. Pick date in the past.	Date: 1/4/2024 Today: 4/4/2024	Notify: please select the future date	Notify: please select the future date	Pass
	FT-012	Create an area to submit assignments.	Student: Salah(spinner with list tutor's students) Title: write a paragraph describing mother Dateline: 7/4/2024	Notify: done And display list an areas to submit assignments	Notify: done And display list an areas to submit assignments	Pass
	FT-013	Update an area to submit assignments. View before edit	Student: Salah(spinner with list tutor's students) Title: write a paragraph describing mother Dateline: 7/4/2024	Display: Student: Salah Title: write a paragraph describing mother Dateline: 7/4/2024	Display: Student: Salah Title: write a paragraph describing mother Dateline: 7/4/2024	Pass
	FT-014	Update an area to submit assignments. Not filling in the input and clicking the ADD button.		Warning for filling in missing.	Warning for filling in missing.	Pass
	FT-015	Update an area to submit assignments. Pick date in the past.	Date: 1/4/2024 Today: 4/4/2024	Notify: please select the future date	Notify: please select the future date	Pass
	FT-016	Update an area to submit assignments.	Student: Salah (spinner with list tutor's students) Title: write a paragraph describing father Dateline: 10/4/2024	Display: Student: Salah Title: write a paragraph describing father Dateline: 10/4/2024	Display: Student: Salah Title: write a paragraph describing father Dateline: 10/4/2024	Pass
	FT-017	Delete an area to submit assignments.	Student: Salah (spinner with list tutor's students) Title: write a paragraph describing father Dateline: 10/4/2024	That area to submit assignments does not exist on screen nor on database	That area to submit assignments does not exist on screen nor on database	Pass
	FT-018	Create tuition.		Warning for filling in missing.	Warning for filling in missing.	Pass

		Not filling in the input and clicking the ADD button.				
	FT-019	Create tuition. Pick date in the past.	Date: 1/4/2024 Today: 4/4/2024	Notify: please select the future date	Notify: please select the future date	Pass
	FT-020	Create tuition.	Student: nunez (spinner with list tutor's students) Amount: 12 Price: 12 Dateline: 20/04/2024 (picker)	Notify: done Display: Student: nunez Amount: 12 Price: 12 Dateline: 20/04/2024 Total: 144 (amount*price)	Notify: done Display: Student: nunez Amount: 12 Price: 12 Dateline: 20/04/2024 Total: 144	Pass
	FT-021	Update tuition. View before edit	Student: Salah Amount: 12 Price: 12 Dateline: 20/04/2024	Student: Salah Amount: 12 Price: 12 Dateline: 20/04/2024	Student: Salah Amount: 12 Price: 12 Dateline: 20/04/2024	Pass
	FT-021	Update tuition. Not filling in the input and clicking the ADD button.		Warning for filling in missing.	Warning for filling in missing.	Pass
	FT-022	Update tuition. Pick date in the past.	Date: 1/4/2024 Today: 4/4/2024	Notify: please select the future date	Notify: please select the future date	Pass
	FT-023	Update tuition.	Student: Salah Amount: 14 Price: 16 Dateline: 20/04/2024	Notify: update success Display: Student: Salah Amount: 14 Price: 16 Dateline: 20/04/2024 Total: 224	Notify: update success Display: Student: Salah Amount: 14 Price: 16 Dateline: 20/04/2024 Total: 224	Pass
	FT-024	Delete tuition	Student: Salah Amount: 14 Price: 16 Dateline: 20/04/2024	That tuition does not exist on screen nor on database	That tuition does not exist on screen nor on database	Pass
	FT-025	Create exam.		Notify : Quiz title can not be empty	Notify : Quiz title can not be empty	pass

		Not filling in the input and clicking the create button.				
	FT-025	Create exam.	Title: Math	Open page add question and answer (multiple choice)	Open page add question and answer (multiple choice)	Pass
	FT-026	Create questions and answers	Tick the missing number [2, 3, 4, 5, ? , 7]: (a) 5 (b) 1 (c) 6 (d) 0 Tick the smallest number (a) 85 (b) 76 (c) 89 (d) 65	Notify: id of exam: 100001	Notify: id of exam: 100001	Pass
	FT-027	View result of exam		List Students take that exam and result's that student	List Students take that exam and result's that student	pass
	FT-028	Chat with tutor's students		Tutor and student chat together	Tutor and student chat together	Pass
	FT-029	Call for tutor's students or parent' student		Tutor call for tutor's students or parent' student by phone number	Tutor call for tutor's students or parent' student by phone number	Pass
	FT-030	Create schedule. Not filling in the input and clicking the create button.		Notify: enter time	Notify: enter time	Pass
	FT-031	Create schedule.	Student: nunez Time: 1:30 CH	Notify: done (save to database)	Notify: done (save to database)	Pass
	FT-032	View schedule.		Display list schedule	Display list schedule	Pass
	FT-033	Update schedule: Not filling in the input and clicking the create button.		Notify: enter time	Notify: enter time	Pass
	FT-031	Update schedule.	Student: nunez Time: 5:30 CH	Notify: update success Display: Student: nunez Time: 5:30 CH	Notify: update success Display: Student: nunez Time: 5:30 CH	Pass

	FT-031	Update schedule. View before update	Student: nunez Time: 1:30 CH	Student: nunez Time: 1:30 CH	Student: nunez Time: 1:30 CH	Pass
	FT-032	Delete schedule:	Student: nunez Time: 1:30 CH	That schedule does not exist on screen nor on database	That schedule does not exist on screen nor on database	Pass

Table 10: test case

Test	Number of test	Pass	Fail	Not yet	No test
Total	82	81	1	0	0

Table 11: result test case

8 Conclusion:

In the TutorKit project, we have successfully developed an application tailored to the needs of both Tutors and Students, as outlined in the requirements. Initially, we created a user-friendly platform that offers a seamless experience for both roles, incorporating essential features and enhancements to enrich the user interaction.

Key features like user registration, login functionality, and profile management were implemented to ensure a personalized experience for both Tutors and Students. Additionally, the system allows for student-tutor matching, class schedule management, and assignment submissions, enhancing the educational process and fostering effective communication between Tutors and Students.

Authentication, Authorization, and Accounting mechanisms were integrated to safeguard the data integrity of the application. This ensures that user data remains secure and accessible only to authorized individuals, enhancing overall system reliability.

To further enhance user experience, we incorporated features like tuition management, test creation, and scores management for Tutors, while Students can conveniently pay tuition, view grades, and take tests. Moreover, we introduced functionalities such as class schedules and assignment submissions to facilitate effective learning.

While TutorKit offers a robust platform with a range of features, there are areas that require attention in future versions. Some functionalities, like real-time updates, could enhance user

engagement by reducing the need for page reloads. Additionally, features like batch data import, more intuitive UX designs, and clearer terms and conditions are essential to improve system usability and compliance.

To address these areas in future versions, we plan to implement real-time features using suitable technologies, integrate batch data import capabilities, enhance UX with more user-friendly designs, and clearly define terms and conditions to ensure user awareness and compliance.

Pros:

- Successfully implemented core features as per requirements.
- Developed a user-friendly platform catering to the needs of both Tutors and Students.
- Enhanced data security and integrity with robust authentication and authorization mechanisms.

Lessons Learned:

- Acquired deeper knowledge in web application development, database management, and user experience design.
- Explored various technologies and frameworks to optimize application performance and functionality.

In conclusion, TutorKit serves as a valuable tool for Tutors and Students alike, facilitating effective communication, learning, and collaboration. With ongoing improvements and feature enhancements, we aim to make TutorKit a leading platform in the educational sector, providing unparalleled support to Tutors and Students in their educational journey.

9 References

9, L., 2024. *LDPlayer 9.* [Online]
Available at: https://ldplayer-9.en.uptodown.com/windows#:~:text=LDPlayer%209%20is%20an%20emulator,with%20better%20quality%20and%20performance
[Accessed 20 04 2024].

abhiandroid, 2024. *JAVA For Android – Tutorial, Examples And Programs.* [Online]
Available at: https://abhiandroid.com/java/#gsc.tab=0
[Accessed 20 04 2024].

Ahmed, A. J., 2024. *Hybrid Power: Flutter Advantages and Benefits.* [Online]
Available at: https://www.toptal.com/flutter/hybrid-power-flutter-advantages
[Accessed 20 04 2024].
Bhatt, T., 2024. *7 Best iOS App Development Programming Languages In 2024.* [Online]
Available at: https://www.intelivita.com/blog/iphone-app-development-languages/#:~:text=Performance%3A%20Similar%20to%20Objective%2DC,Swift%20projects%20and%20vice%20versa
[Accessed 20 04 2024].
Bui, H., 2021. *Flutter App Development: Pros, Cons, Characteristics and More.* [Online]
Available at: https://wearefram.com/blog/flutter-mobile-app-development/
[Accessed 20 04 2024].
bvop, 2023. *What is Scope Management in Agile Project Management.* [Online]
Available at: https://bvop.org/learn/scopemanagement/#:~:text=Agile%20Scope%20Management%20suggests%20that,opportunity%20rather%20than%20a%20problem.
[Accessed 20 11 2023].
canvas, 2024. *OVERVIEW.* [Online]
Available at: https://www.trustradius.com/products/canvas/reviews
[Accessed 20 04 2024].
codementor, 2023. *Swift Package Manager vs CocoaPods vs Carthage for All Platforms.* [Online]
Available at: https://www.codementor.io/blog/swift-package-manager-5f85eqvygj
[Accessed 20 04 2024].
developer, 2024. *Meet Android Studio.* [Online]
Available at: https://developer.android.com/studio/intro
[Accessed 20 04 2024].
Edmodo, 2024. *OVERVIEW.* [Online]
Available at: https://www.trustradius.com/products/edmodo/reviews#overview
[Accessed 20 04 2024].
firebase, 2023. *Choose a Database: Cloud Firestore or Realtime Database.* [Online]
Available at: https://firebase.google.com/docs/database/rtdb-vs-firestore
[Accessed 20 11 2023].
firebase, 2023. *Firebase Realtime Database.* [Online]
Available at: https://firebase.google.com/docs/database
[Accessed 20 11 2023].
firebase, 2023. *Firebase Security Rules.* [Online]
Available at: https://firebase.google.com/docs/rules
[Accessed 20 11 2023].
flutter, 2024. *FAQ.* [Online]
Available at: https://docs.flutter.dev/resources/faq
[Accessed 20 04 2024].
Gupta, E., 2024. *Why Java is Platform Independent?.* [Online]
Available at: https://www.shiksha.com/online-courses/articles/why-java-is-platform-independent-blogId-160499#:~:text=Java%20achieves%20platform%20independence%20through,underlying%20ha

rdware%20or%20operating%20system
[Accessed 20 04 2024].
Krysik, A., 2024. *Is Java Dead in 2024? The Truth About Java Popularity.* [Online]
Available at: https://stratoflow.com/why-is-java-so-popular/
[Accessed 20 04 2024].
leadschool11, 2023. *Student Management System - Key Features & Benefits.* [Online]
Available at: https://issuu.com/leadschool11/docs/student_management_system_-_key_features_benefit
[Accessed 14 10 2023].
Learning, S., 2024. *OVERVIEW.* [Online]
Available at: https://www.trustradius.com/products/powerschool-schoology-learning/reviews
[Accessed 20 04 2024].
Lido, 2023. *What Type of Database is Firebase? (2023 Update).* [Online]
Available at: https://www.lido.app/firebase/what-type-of-database-is-firebase#:~:text=Firebase%20offers%20two%20types%20of,different%20use%20cases%20and%20requirements.
[Accessed 20 11 2023].
logixbuilt, 2024. *Building Cross-Platform Apps with Flutter.* [Online]
Available at: https://logixbuilt.com/building-cross-platform-apps-with-flutter/
[Accessed 20 04 2024].
microsoft, 2021. *Data types.* [Online]
Available at: https://learn.microsoft.com/en-us/dotnet/standard/data/sqlite/types
[Accessed 20 11 2023].
Mirzajanzadeh, A., 2024. *The Java Ecosystem: Libraries and Frameworks Every Developer Should Know.* [Online]
Available at: https://www.linkedin.com/pulse/java-ecosystem-libraries-frameworks-every-developer-ali-mirzajanzadeh
[Accessed 20 04 2024].
Nayak, S. K., 2023. *How is Flutter Able to Achieve Native Performance on Different Platforms?.* [Online]
Available at: https://www.linkedin.com/pulse/how-flutter-able-achieve-native-performance-different-nayak
[Accessed 20 04 2024].
Raj, R. S., 2024. *Ahead-of-Time Compilation vs. Just-in-Time Compilation in Java: A Comparative Analysis.* [Online]
Available at: https://www.linkedin.com/pulse/ahead-of-time-compilation-vs-just-in-time-java-comparative-raj
[Accessed 20 04 2024].
sqlite, 2023. *SQL As Understood By SQLite.* [Online]
Available at: https://www.sqlite.org/lang.html#:~:text=SQL%20As%20Understood%20By%20SQLite,of%20the%20standard%20SQL%20language.
[Accessed 20 11 2023].

sqlite, 2023. *sqlite - Sqlite Web Security.* [Online]
Available at: https://www2.sqlite.org/cvstrac/wiki?p=SqliteWebSecurity
[Accessed 20 11 2023].
swift, 2024. *About Swift.* [Online]
Available at: https://www.swift.org/about/
[Accessed 20 04 2024].
swift, 32024. *Platform Support.* [Online]
Available at: https://www.swift.org/platform-support/
[Accessed 20 04 2024].
Tillu, J., 2023. *What is Swift Language?.* [Online]
Available at: https://www.linkedin.com/pulse/what-swift-language-jay-tillu-uydyf#:~:text=Swift%20has%20gained%20popularity%20among,projects%20like%20SwiftNIO%20and%20Vapor
[Accessed 20 04 2024].
trello, 2024. *Learn Trello board basics.* [Online]
Available at: https://trello.com/guide/trello-101
[Accessed 20 04 2024].
tutorialspoint, 2023. *SDLC - Agile Model.* [Online]
Available at: https://www.tutorialspoint.com/sdlc/sdlc_agile_model.htm
[Accessed 20 11 2023].
tutorialspoint, 2023. *SDLC - Waterfall Model.* [Online]
Available at: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm
[Accessed 11 20 2023].
usemotion, 2023. *Understanding the Waterfall Methodology: A Sequential Approach to Project Management.* [Online]
Available at: https://www.usemotion.com/blog/waterfall-methodology
[Accessed 20 11 2023].
Waseem, A., 2023. *Waterfall Methodology: History, Principles, Stages & More.* [Online]
Available at: https://management.org/waterfall-methodology#:~:text=Minimal%20customer%20involvement%3A%20A%20waterfall,and%20objectives%20are%20clearly%20defined.
[Accessed 20 11 2023].