National University of computer and emerging sciences

Final Project

CS-328 Software Engineering

BS(CS) – D

Batch 2018

Submitted By:

Hassan Shahzad	18i-0441
Azka Khurram	18i-0461
Abeera Fatima	18i-0411
Sana Ali	18i-0439

Submitted to:

Maam Maryam Abdul Ghafoor

Date of Submission:

16-06-21

PHY, A WALK-THROUGH PHYSICS – LEARN ONLINE APPLICATION

CONTENTS

INTRODUCTION

'*Phy, a walkthrough physics*' is a learn-online app designed for our client, which aims to help him organize and distribute his learning resources to his students through a single source (the application). Phy will be linked to the client's Google drive – It is currently in its last phase of production. We are just awaiting the client's approval before finalizing the product – from where it will fetch the links of the respective resource for the students and store it in its firebase database. The students will be able to access the latest material through the app.

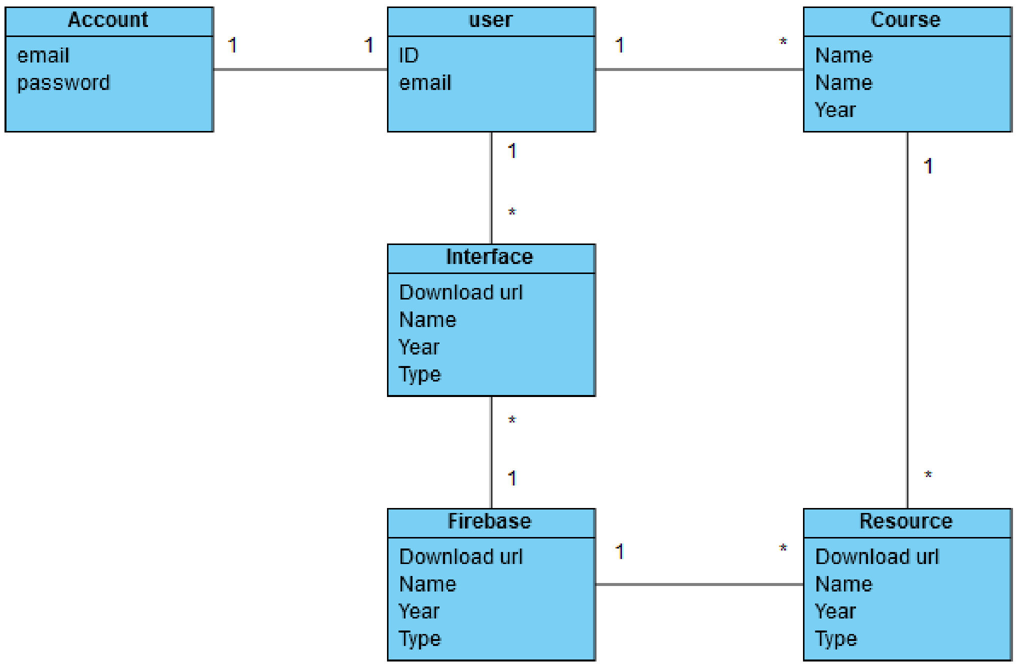
SOFTWARE ARCHITECTURE

RATIONALE:

We have chosen to implement our software with a layered architecture, as it is easy to understand and helps with the division of work, for e.g., we had separate group members tackling the interface, the logical layer and the database simultaneously (a feature extremely useful due to limited working time). This type of architecture, organizes the code in an easily editable manner. Someone else, who plans to change the database for example, can easily locate the respective files and make their desired changes without disturbing the rest of the code, which does not concern them. This type of architecture is, therefore, allowing us to separately test the modules, and make required changes without having to introduce changes to the rest of our functionality. Thus, in case of failure, the project is easy to debug and an error in one layer will not be translated to other layers. Since we plan on releasing this app in the future, having an interface which keeps the UI and the backend code separate is ideal for us as we had these layers developed by separate members, so each person can continue to work on their parts without interference from the others.

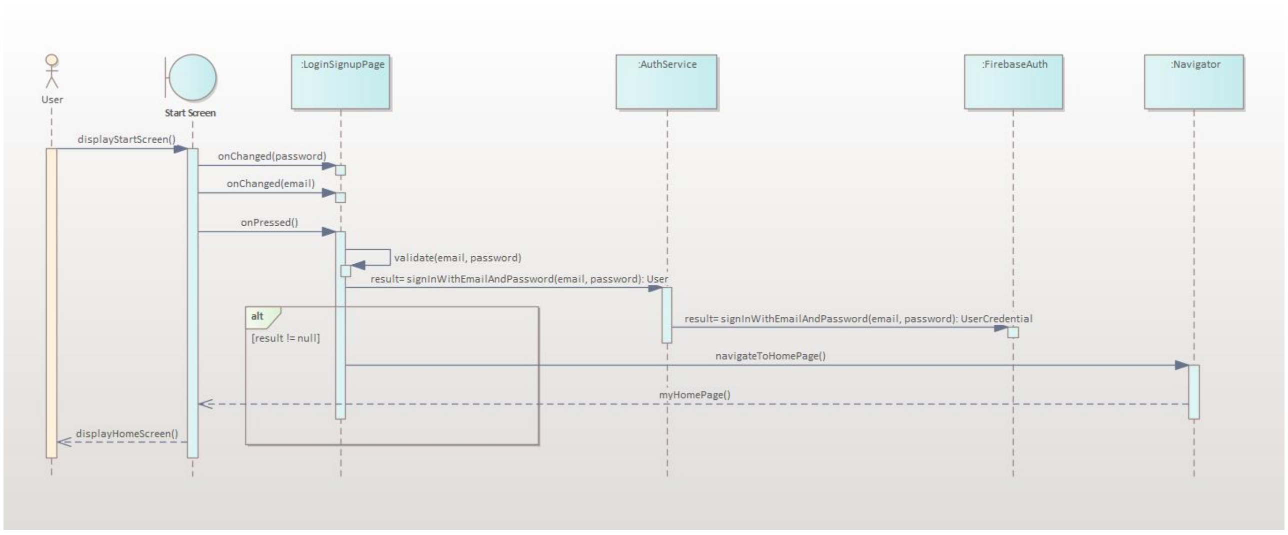
However, this architecture did introduce some issues: our app's performance in terms of speed is slightly lower than what we had anticipated, and adding a new module will require changes to be made to all layers individually too. We also noticed that as we were programming in dart, it was hard to differentiate between the Business logic and the Presentation layer, whilst maintaining a good efficiency in terms of space. Sometimes there was relatively very little code to be dealt with. Making separate functions and classes for dealing with some logical calculations seemed to be a waste of space. Such was also the case with the Firebase integration with the Business layer. Some in built functionality was achieving the same purpose in lesser code than when we separated the code into separate layers.

CLASS DIAGRAM:

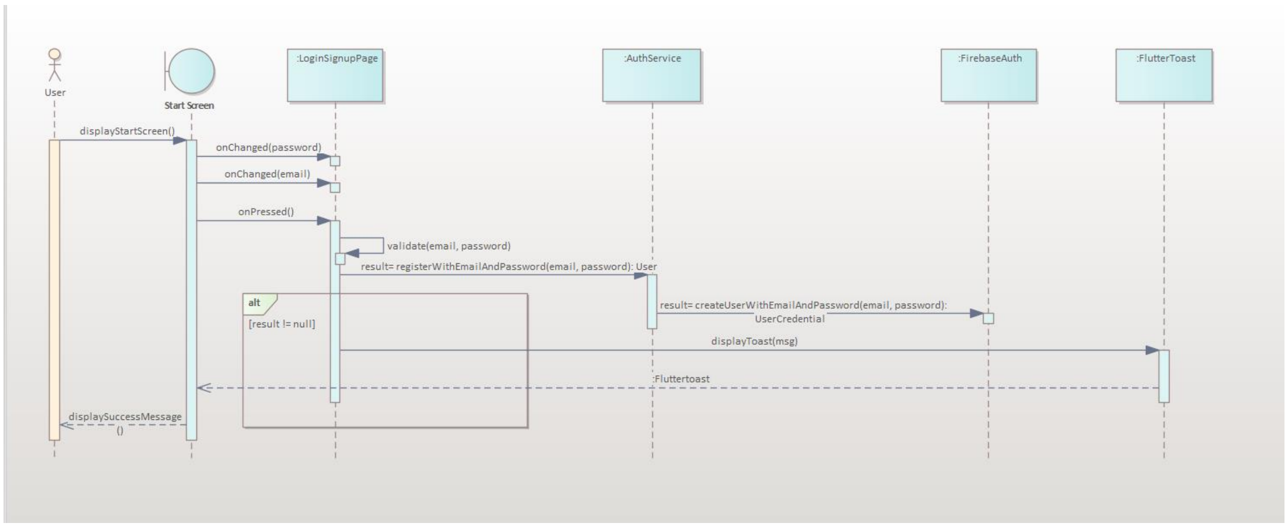


SEQUENCE DIAGRAMS:

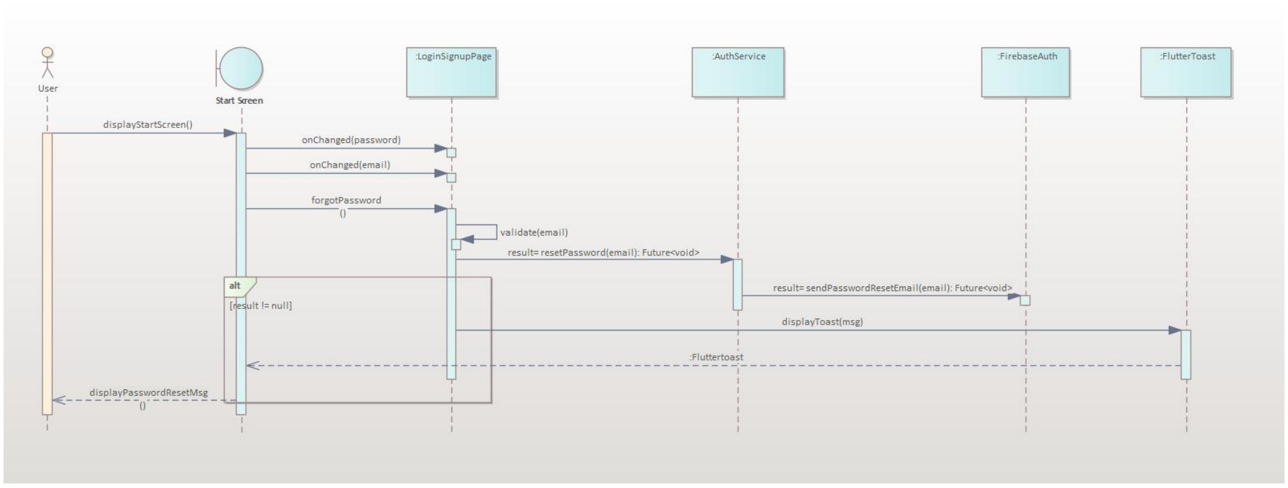
LOGIN:



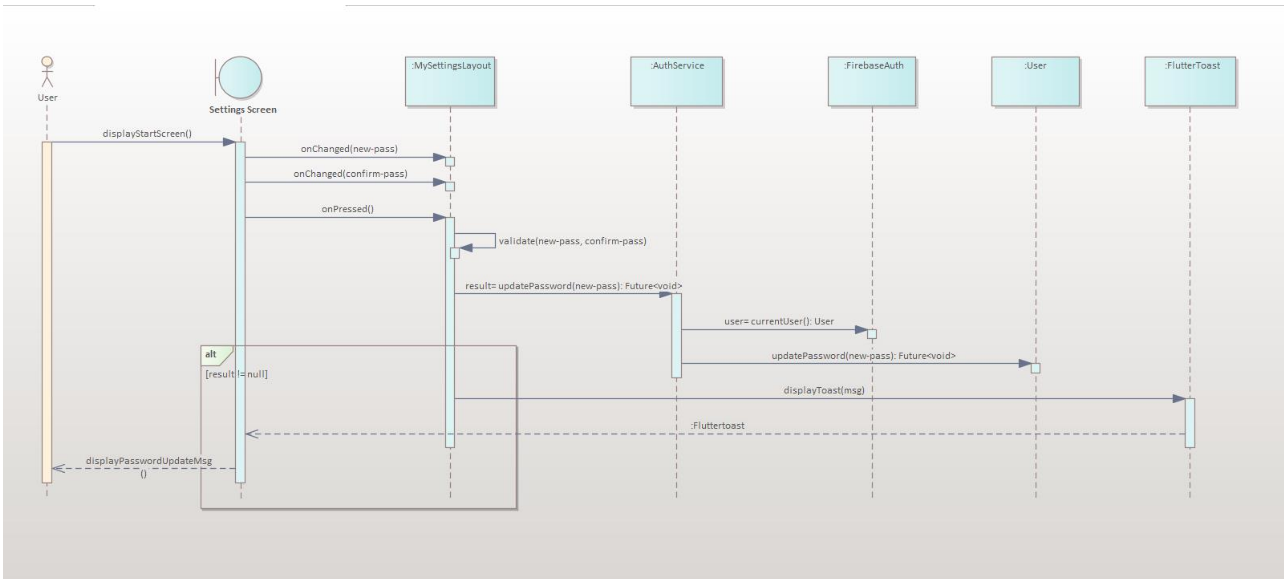
SIGNUP:



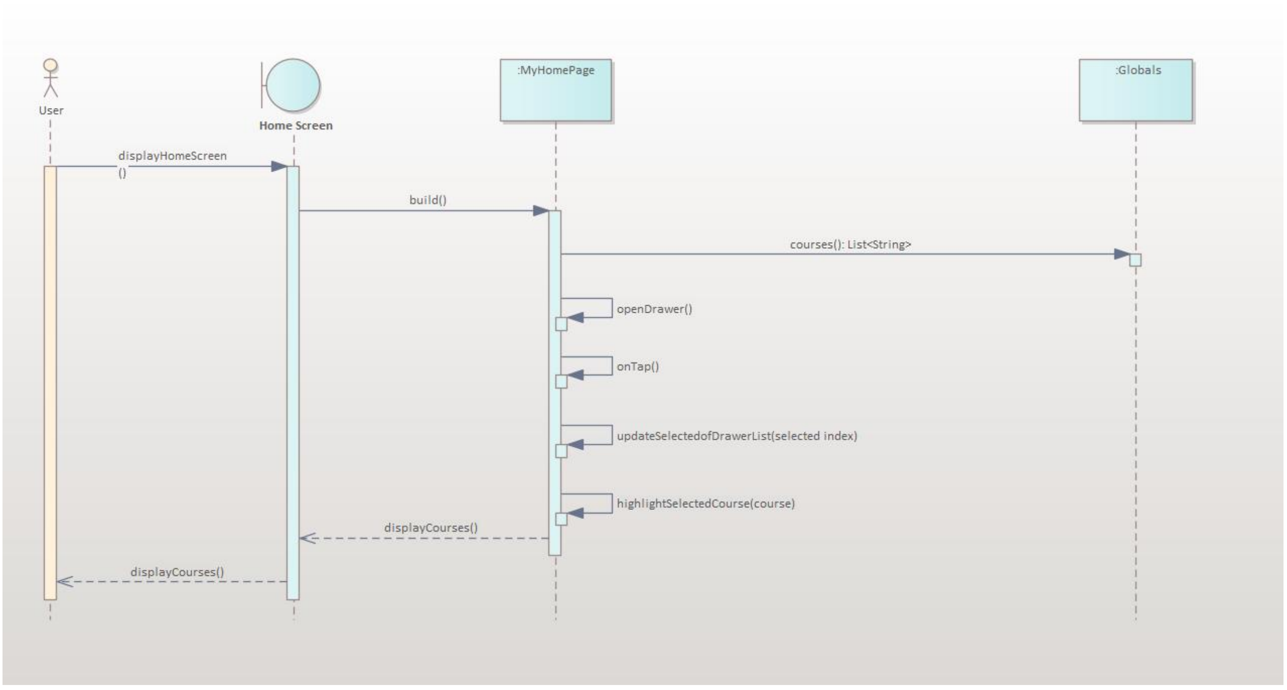
ACCOUNT RECOVERY:



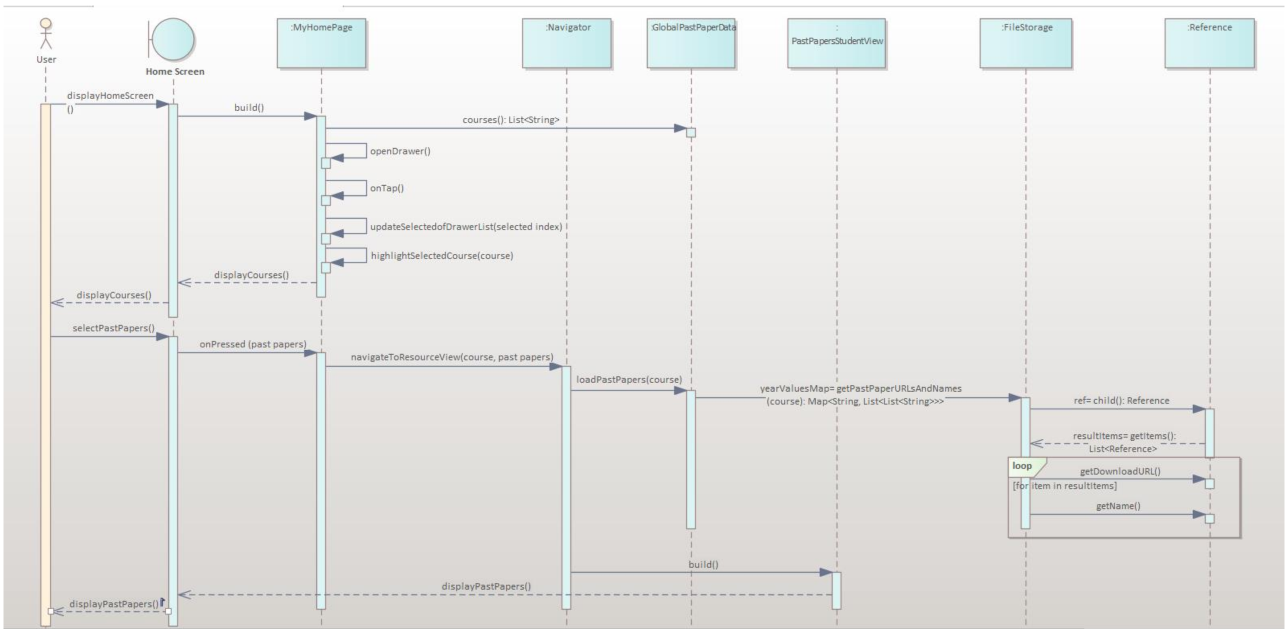
CHANGE PASSWORDS:



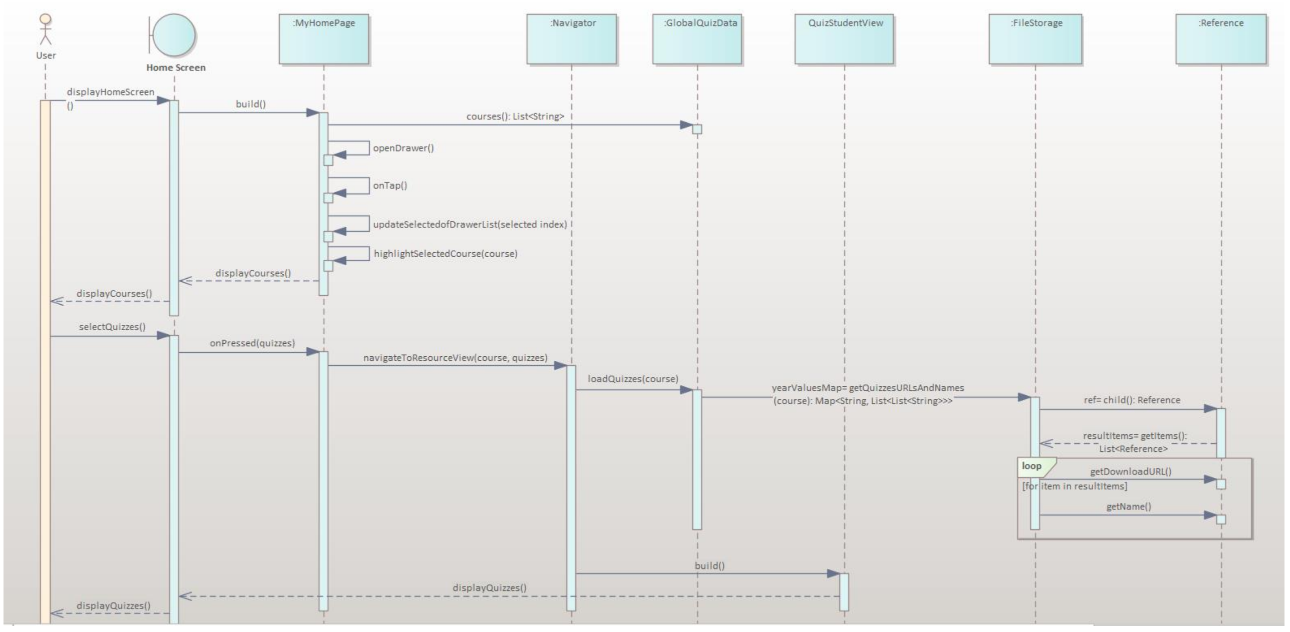
VIEW COURSES:



VIEW PAST PAPERS:

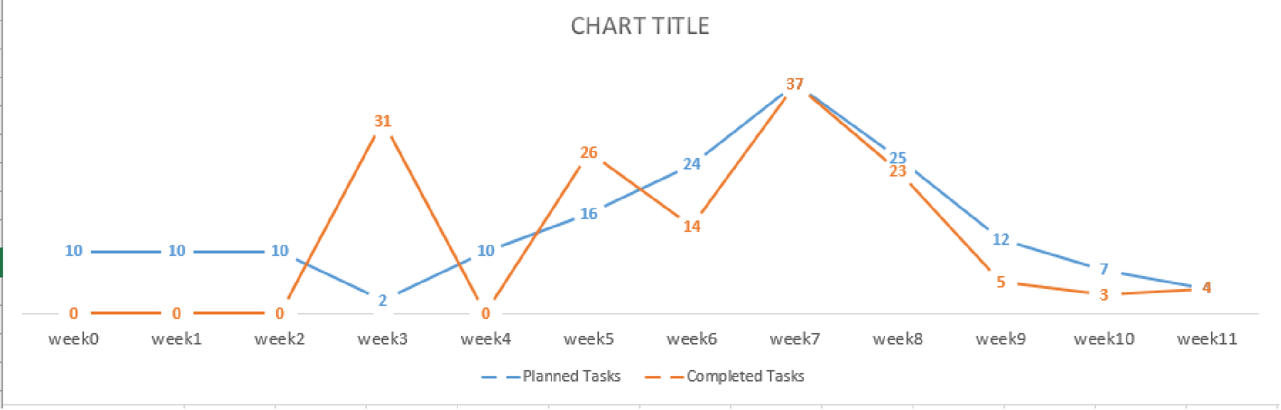


VIEW QUIZZES:



BURN-DOWN CHART

	week0	week1	week2	week3	week4	week5	week6	week7	week8	week9	week10	week11
Planned Tasks	10	10	10	2	10	16	24	37	25	12	7	4
Completed Tasks	0	0	0	31	0	26	14	37	23	5	3	4



PRODUCT REVIEW

POSSIBLE IMPROVEMENTS:

1. Improve the data fetch and load speed. The app takes approximately 5 seconds to load the data, which can be irritating for some users who want to rapidly switch between the modules.
2. We can save the state of the user so they do not have to repeatedly login every time into the app. Currently this functionality is not available.
3. Administrative control can be given to the client so he can make his desired changes directly in the app (rather than having to go through us.)
4. Chat boxes can be added for students to directly communicate with their instructor.
5. Comment sections can be added under the resources for students to converse related to the respective resource.

LIMITATIONS AND CONSTRAINTS:

1. If the client wants to make any changes to the application, he will have to make them through the development team (us), as currently no administrative control is given to the client.
2. The application is specifically designed for android devices, iOS phones and Windows devices cannot use it.
3. The oldest android version supported by the device is kit-kat.

INITIAL PRODUCT BACKLOG:

User story	Time Estimated	Priority
As a student, I want to view courses list so I may be able to register for the course.	5 hours	4
As a student, I want to view video lectures so I may be able to study for the course.	3 hours	3
As a student, I want to view past papers so I may be able to select the past paper I want	2 hours	3
As a user, I want to be able to register my account with Physics 101 so that I can access the system.	7 hours	5
As a user, I want to be able to log in so that I can use the system	5 hours	5
As a user, I want to be able to recover my account in case I forget my password so that I can access app with that account.	10 hours	3
As a user, I want to be able to change my password so that I can pick a new password if I want.	6 hours	3
As a user, I want to be able to view the syllabus of a course so that I can view the topics and chapters.	2 hours	3
As a user, I want to be able to view quizzes of a course so that I can select and view the quiz relevant to my interest.	1 hour	3
As a user, I want to be able to view worksheets of a course so that I can select and view the worksheet relevant to my interest.	1 hour	3
As the teacher, I want to be able to add video lectures so I that I can store more lectures in the app	4 hours	2
As the teacher, I want to be able to add past papers so I that I can store more past papers in the app	4 hours	2
As the teacher, I want to be able to add the syllabus so I that I can store more syllabi in the app	4 hours	1
As the teacher, I want to be able to add quizzes so I that I can store more quizzes in the app	4 hours	2
As the teacher, I want to be able to add worksheets so I that I can store more worksheets in the app	4 hours	2
As a user, I want to be able to edit my account info so that I can update my account details	2 hours	3
As a teacher, I want to able to view the number of students registered in a course so I may know how many students take each course	1 hour	1

CHANGES MADE TO THE INITIAL PRODUCT BACKLOG:

Initially we planned to give more control to our client in terms of editing, creating and deleting any resources. However, this idea was dropped after our last meeting with our client when we realized that they were struggling whilst using their google drive and were not clear about their own preferences or requirements. Thus, we decided to perform the crud operations ourself. The client will be able to email us through the app with their desired changes and we will update the database from our end, which will automatically change the app as our interface is dynamically populated.

We might change this mode of operation in the future, however, depending on the final meeting we will have with the client shortly.

FINAL PRODUCT BACKLOG:

User story	Replaced by – user story	Reason for change	Tasks	Developed in sprint	Developed by	Time taken for development
As a student, I want to view courses list so I may be able to register for the course.	Not changed	-	• Home Page Class • Database Methods • Making Custom Classes • Global Variables Class • Functions to store and pass selected course • Linking from home screen • Dynamically loading courses	2	Abeera Fatima, Sana Ali	5 hours
As a student, I want to view video lectures so I may be able to study for the course.	Not changed	-	• Make display for menu • Switch between menu and lectures screen • Display list of lectures from Database • Link with Login Screen	2	Azka Khurram, Hassan Shahzad, Sana Ali	3 hours

			• Build lectures collection in database			
As a student, I want to view past papers so I may be able to select the past paper I want	Not changed	-	• Make display for menu • Switch between menu and lectures screen • Display list of past papers from Database • Link with Login Screen • Build past papers collection in database	2	Azka Khurram, Hassan Shahzad, Sana Ali	2 hours
As a user, I want to be able to register my account with Physics 101 so that I can access the system.	Not changed	-	• Home Page Class • Database Class • Authentication Class • Making Custom Widgets • Linking App to Firebase • Successful Access to Data • Linking screen to login • Defined method for adding account to Firebase • Homepage Class	Completed in sprint 1, reworked in sprint 2	Abeera Fatima, Sana Ali, Hassan Shahzad	7 hours
As a user, I want to be able to log in so that I can use the system	Not changed	-	• Home Page Class • Database Class • Authentication Class • Making Custom Widgets • Linking App to Firebase	Completed in sprint 1, reworked in sprint 2	Abeera Fatima, Hassan Shahzad	5 hours

			• Successful Access to Data • Linking screen to login • Defined method for logging in to Firebase • Homepage Class			
As a user, I want to be able to recover my account in case I forget my password so that I can access app with that account.	Not changed	-	• Button to forget password • Function to link to database • Authorization Class • Error Styling • Testing	2	Abeera Fatima, Sana Ali, Hassan Shahzad	10 hours
As a user, I want to be able to change my password so that I can pick a new password if I want.	Not changed	-	• Setting Class • Function to link to database • Authorization Class • Error Styling • Disable email edit • Functions to get current user • Testing	2	Abeera Fatima, Sana Ali, Hassan Shahzad	6 hours
As a user, I want to be able to view the syllabus of a course so that I can view the topics and chapters.	Not changed	-	• Syllabus Class • Database Methods • Global Variables Class • Making Custom Widgets • Function to get syllabus by course • Loading files from database • Function for opening syllabus in browser	2	Azka Khurram, Hassan Shahzad, Sana Ali	2 hours

As a user, I want to be able to view quizzes of a course so that I can select and view the quiz relevant to my interest.	Not changed	-	• Quiz Class • Database Methods • Global Variables Class • Making Custom Widgets • Function to generate dynamic list of chapters • Item Class to make Tiles • Displaying quizzes per chapter • Storage Class • Loading Files from database • Testing with various courses and chapters	2	Azka Khurram, Hassan Shahzad, Sana Ali	1 hour

SCRUM RETROSPECTIVE

WORK DIVISION PER SPRINT:

Each sprint was divided by keeping in mind the doability of the assigned tasks in the given time period. Since we were unfamiliar with the language that we were using, we only implemented the login, signup and main page in our first sprint. Once we got familiar with what we were using, we redid the tasks of the first sprint along with some other major functionality of our project. We managed the tasks in such a way that we will not have much work for sprint three as there wasn't a lot of time allotted for it.

PROCESS (HOW THE WORK WAS CARRIED OUT):

The team was further divided into two teams. One team was responsible for the frontend and the second team was responsible for the backend. Tasks were assigned to each team member during the start of every sprint. Since the tasks completed by the frontend team depended upon the tasks completed by the backend team, time was allocated in such a way that there would be sufficient time for both teams to carry out their assigned tasks.

POSSIBLE IMPROVEMENTS:

The set deadlines should have been followed strictly.

POSSIBLE IMPROVEMENTS IN THE TEAM MANAGEMENT FOR FUTURE PROJECTS:

Communication process between the team members could have been improved. Before implementation of any task team members should have been informed.

DIFFICULTIES FACED AND HOW THEY WERE HANDLED:

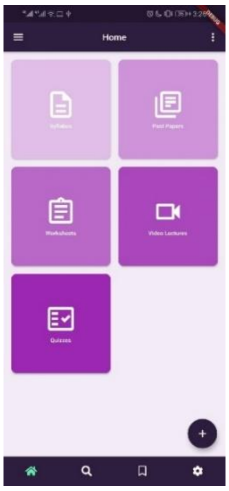
Our biggest issue was time management. In sprint one; tasks were not finished in the assigned time period. For the second sprint, we decided to complete almost all the tasks so that we won't have much work to do in the third sprint. Our client changed his requirements during the middle of our working process. We removed those parts from our product backlog.

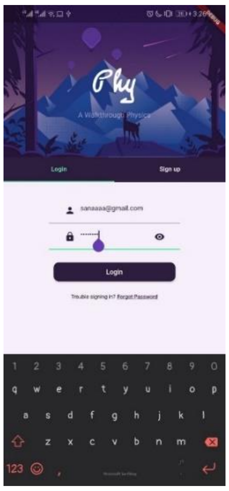
LESSONS LEARNED:

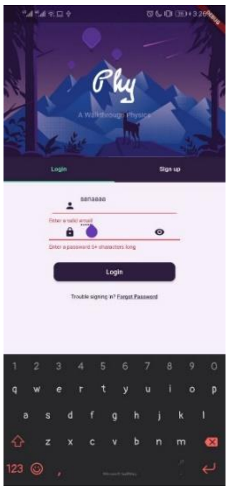
Time management is very important in such projects. We should have been able to manage our project along with our other work. It should have been communicated to the client that requirements cannot be changed in the middle of the project process.

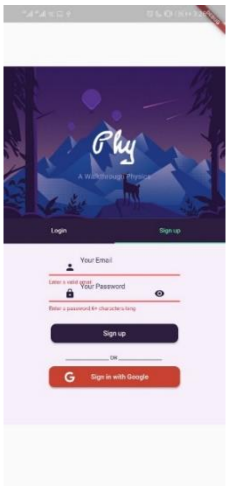
PROJECT INTERFACE:

SCREENSHOTS:






Phy
A Walkthrough: Physics
Login
Sign up
Enter a valid email
Enter a password 6+ characters long
Login
Trouble signing in? Forgot Password



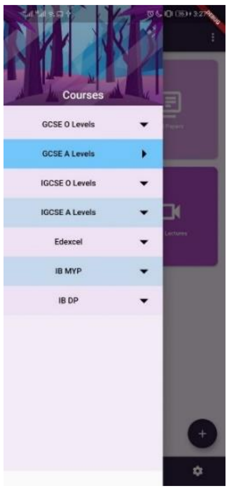
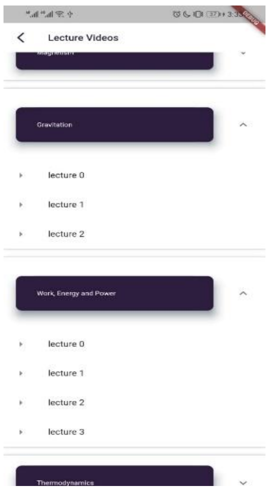
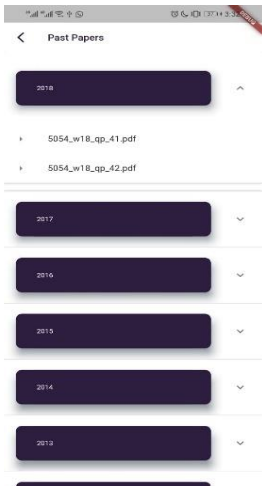
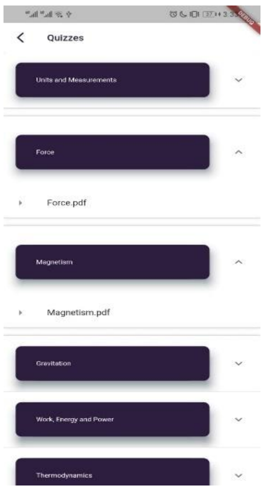


Fig1: User Home Screen Fig2: Login Screen 1 Fig3: Login Screen Fig4: Signup Screen Fig5: Menu Bar







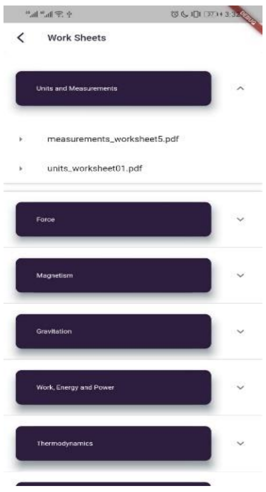
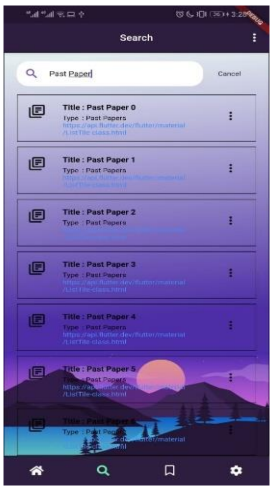
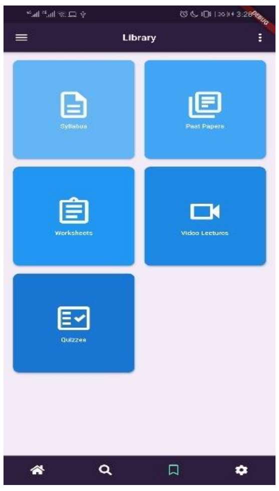
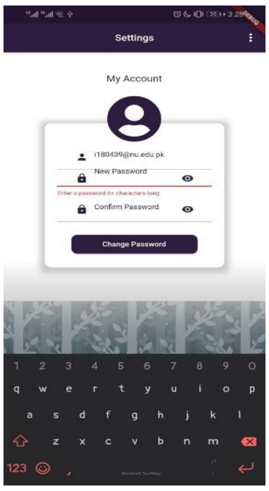


Fig6: View Lectures Fig7: View Past papers Fig8: View Quizzes Fig9: View Worksheets







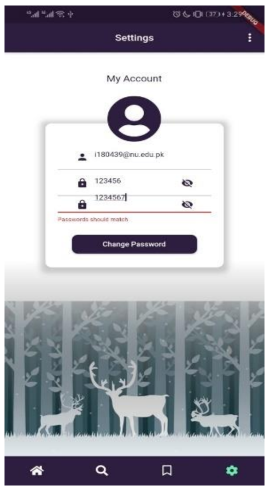
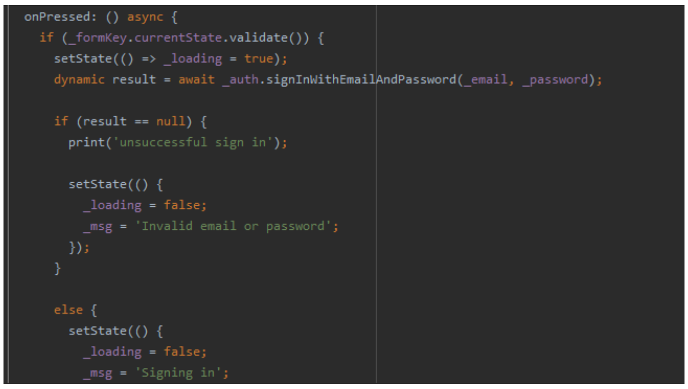


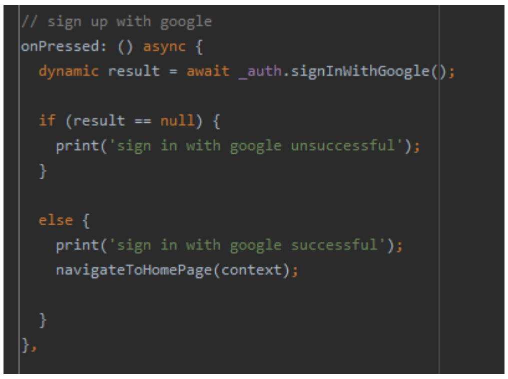
Fig10: Search Past Papers Fig11: Saved Resources' Screen Fig12: Account Settings Fig13: Passwords don't match

BACKEND:



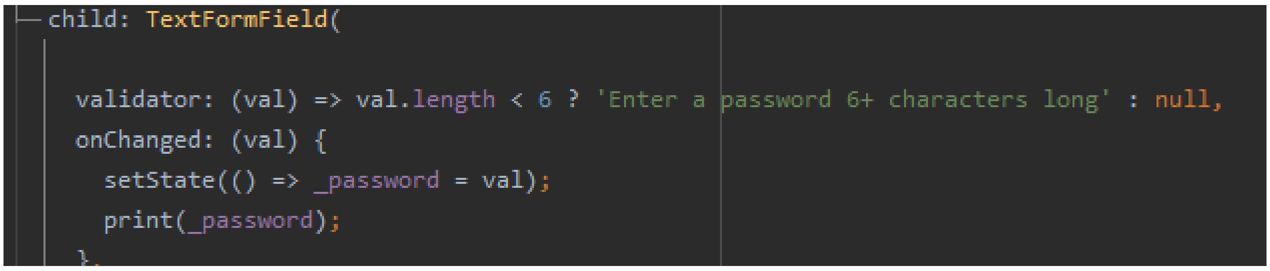
```
onPressed: () async {
  if (_formKey.currentState.validate()) {
    setState(() => _loading = true);
    dynamic result = await _auth.signInWithEmailAndPassword(_email, _password);

    if (result == null) {
      print('unsuccessful sign in');

      setState(() {
        _loading = false;
        _msg = 'Invalid email or password';
      });
    }

    else {
      setState(() {
        _loading = false;
        _msg = 'Signing in';
```

Fig 8.1: Input Validation of Email and Password



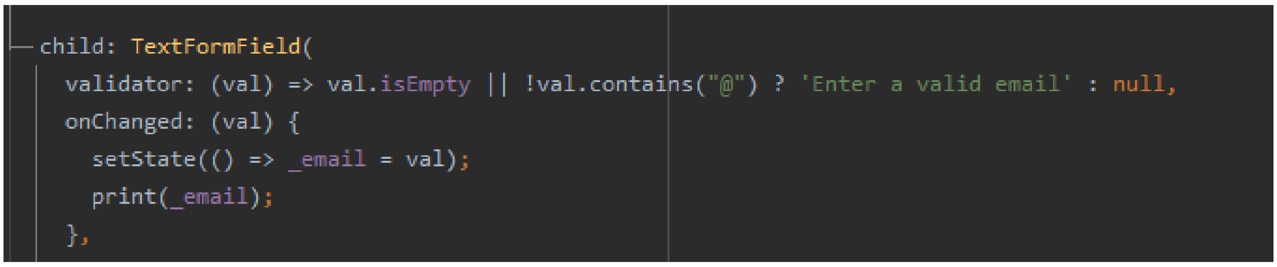
```
// sign up with google
onPressed: () async {
  dynamic result = await _auth.signInWithGoogle();

  if (result == null) {
    print('sign in with google unsuccessful');
  }

  else {
    print('sign in with google successful');
    navigateToHomePage(context);

  }
},
```

Fig 8.2: Validates Google Sign in and then go to Homepage



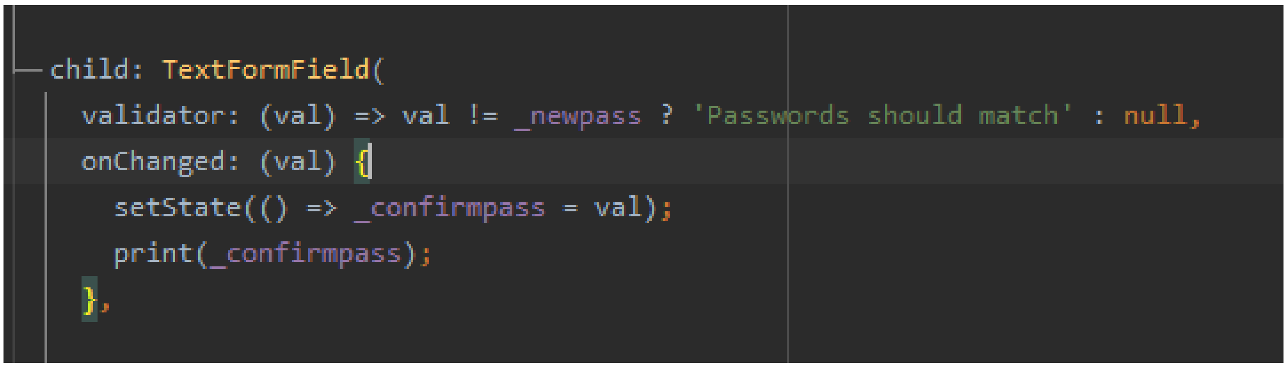
```
child: TextFormField(

  validator: (val) => val.length < 6 ? 'Enter a password 6+ characters long' : null,
  onChanged: (val) {
    setState(() => _password = val);
    print(_password);
  },
```

Fig 8.3: Password Validation



```
child: TextFormField(
  validator: (val) => val.isEmpty || !val.contains("@") ? 'Enter a valid email' : null,
  onChanged: (val) {
    setState(() => _email = val);
    print(_email);
  },
```

Fig 8.4: Email Validation



```
child: TextFormField(
  validator: (val) => val != _newpass ? 'Passwords should match' : null,
  onChanged: (val) {
    setState(() => _confirmpass = val);
    print(_confirmpass);
  },
```

Fig 8.5: Validates that “Confirm Password” is same as “New Password”

TESTING OF GUI FIELDS:

EQUIVALENCE CLASSES:

User Story	Equivalence Classes	Number of test cases for equivalence		Boundary Values
		strong	weak	
As a user I should be able to login	Password {length = 0, 0 < length < 6, length >= 6} Email {no characters, does not include '@' character, includes '@' character}	9	3	Password length = 0 Password length = 6 Email: 0 characters Email: Without '@' character
As a user I should be able to signup	Password {length = 0, 0 < length < 6, length >= 6} Email { no characters, does not include '@' character, includes '@' character }	9	3	Password length = 0 Password length = 6 Email: 0 characters Email: Without '@' character

As a user I should be able to update my password	New Password {length = 0, 0 < length < 6, length >= 6}
	
	Confirm Password {
	Empty, not equal to password, equal to password}
		9	3	New Password length = 0
				New Password length = 6
				
				confirm password != new password

As a user I should be able to recover my account	Email {no characters, does not include '@' character, includes '@' character }	3	3	Email: 0 characters Email: Without '@' character

TEST CASES:

Test Case ID	Test Scenario	Test Steps	Test Data	Valid Classes	Invalid Classes

#1	User Sign up	• Open app to sign up page
		• Leave email and password fields blank
		• Click sign up
			Email = -
			Password = -
				-	Email: {no characters}
					Password: {length = 0}

#2	User Sign up	• Open app to sign up page
		• Enter invalid email and password
		• Click sign up
			Email = 'hellogmail.com'
			Password = '12345'
				-	Email: {does not include '@' character}
					Password: {0 < length < 6}

#3	User Sign up	• Open app to sign up page
		• Enter email and password
		• Click sign up
			Email = 'sana1@gmail.com'
			Password = 'hello123'
				Email: {includes '@' character}
				Password: {length > 6}
					-

#4	Google Sign In	• Open app to sign up page • Click on google sign in • Choose google account to continue with	Google account = 'i180439@nu.edu.pk'	Email: {includes '@' character}	-
#5	Google Sign In	• Open app to sign up page • Click on google sign in • Choose an incorrect	Google account = 'sanayahoo.com'	-	Email: {does not include '@' character}

		google email			

#7	Login	• Open app to login page
		• Enter email and password
		• Click on login
			Email = ‘i180439@nu.edu.pk’
			Password = ‘hello1234’
				Email: {includes ‘@’ character}
				Password: {length > 6}
					-

	Login	• Open app to login page • Enter email and password • Click on login	Email = ‘i180439@nu.edu.pk’ Password = ‘hello’	Email: {includes ‘@’ character}	Password: {0 < length < 6}
#9	Forgot Password	• Open app to login page • Click on forgot password	Email = - Password = -	-	Email: {no characters}
#10	Forgot Password	• Open app to login page • Enter email • Click on forgot password • Open email and click the link sent to generate new password • Use new password to login	Email = ‘i180439@nu.edu.pk’ New password = ‘hello1234’	Email: {includes ‘@’ character}	-

#11	Change Password	• Open app and login
		• Go to settings page
			New password = -
			Confirm password = -
				-	New Password {length = 0}
					Confirm Password {Empty}

		• Click on change password			
#12	Change Password	• Open app and login • Go to settings page • Enter new password and confirm password • Click on change password	New password = ‘123456’ Confirm password = ‘234566y5re43’	New Password {length >= 6}	Confirm Password {not equal to password}

#13	Change Password	• Open app and login
		• Go to settings page
		• Enter new password and same confirm password
		• Click on change password
			New password = ‘hello1234’
			Confirm password = ‘hello1234’
				New Password {length >= 6}
				
				Confirm Password {
				equal to password}
					-

COMPLETE CODE OF THE PROJECT:

The source code is included in the zip folder.