

Name: J.A. Mujeeb

Student Reference Number: 10707284

Module Code: CNET343SL	Module Name: Distributed Systems
Coursework Title: Weather Reporting System Proposal	
Deadline Date: 03/12/2021	Member of staff responsible for coursework: Mr. Pramudya Thilakaratne.
Programme: BSc (Hons) Plymouth Software Engineering	

Please note that University Academic Regulations are available under Rules and Regulations on the University website www.plymouth.ac.uk/studenthandbook.

Group work: please list all names of all participants formally associated with this work and state whether the work was undertaken alone or as part of a team. Please note you may be required to identify individual responsibility for component parts.

J.A. Mujeeb – 10707284
 G.M.D.D. Ratnayake – 10707351
 S.O. Perera – 10707315
 N. S. De Alwis – 10707160
 M. D. A. Medhavi – 10707278
 P. P. L. Dilhani – 10709402

We confirm that we have read and understood the Plymouth University regulations relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.

Signed on behalf of the group: J.A. Mujeeb

Individual assignment: ***I confirm that I have read and understood the Plymouth University regulations relating to Assessment Offences and that I am aware of the possible penalties for any breach of these regulations. I confirm that this is my own independent work.***

Signed:



Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence.

I *have used/not used translation software.

If used, please state name of software.....

Overall mark _____ % Assessors Initials _____ Date _____

- *Please delete as appropriateSci/ps/d:/students/cwkfrontcover/2013/14

Contents

Acknowledgement.....	6
Introduction to the Distributed System.....	6
Introduction to Quick Weather –.....	6
1.1 What is Quick Weather?	7
Usefulness of the application to the real-world.....	7
Scope of Project	7
1.2 Diagrams.....	8
1.2.1 System Diagram	8
1.2.2 Context Diagram.....	9
1.2.3 Use Case Diagram	9
Mobile Application as Client	9
Methodology	10
Database: PhpMyAdmin MySQL.....	10
Gantt Chart.....	11
Selection of Middleware.....	11
Technical Diagram.....	11
1.3 Overview of Technical Diagram.....	11
1.4 Technical Architectural Diagram	12
1.4.1 Overview of Technical Diagram	12
1.4.2 Middleware of Technical Diagram	12
1.4.3 Technical Architectural Diagram	12
Load Balancing.....	13
1.5 With Load Balancing.....	13
1.6 Without Load Balancing.....	13
Tolerance to Network Failure.....	14
System architectural Diagram	14
1.7 User Requirements.....	14
1.8 Functional Requirements –	14
API / Middleware Justification	14
Software components used	15
Issues faced and actions taken	15
Development phases.....	15
Tasks undertaken	16
1.9 Database.....	16
1.10 Postman API requests and testing.....	16
1.10.1 Add Post Request.....	17
1.10.2 Login Request.....	17
1.11 Screenshots of the website.....	17
1.12 Screenshots of the Mobile Application.....	18
1.13 Main codes of the website	20

1.14	Main codes of the mobile application.....	20
1.14.1	Create an account mobile application code	20
1.14.2	Login	21
1.14.3	Dashboard	21
1.14.4	Post records.....	22
1.14.5	Signup Screen	22
1.14.6	Models Folder	22
1.14.7	Post list adapter folder	23
1.15	Main codes of the Desktop application	23
1.16	Main codes of the API	26
1.16.1	Controller	26
1.16.2	APIs	29
Future enhancement of the system	36
Risk assessment	36
Quality plan	37
Summary	37
Individual contribution	37
References	39
 Figure 1:	System Diagram	8
Figure 2:	Context Diagram	9
Figure 3:	Use case Diagram	9
Figure 4:	Gantt Chart (Part 1)	11
Figure 5:	Gantt Chart (Part 2)	11
Figure 6:	Overview of Technical Diagram	11
Figure 7:	Overview of Technical Diagram	12
Figure 8:	Middleware of Technical Diagram	12
Figure 9:	Technical Architectural Diagram	12
Figure 10:	With Load Balancing	13
Figure 11:	Without Load Balance	13
Figure 12:	System Architectural Diagram.....	14
Figure 13:	Database	16
Figure 14:	Stored data of users.....	16
Figure 15:	Post submission.....	16
Figure 16:	Create an account.....	20
Figure 17:	Login code	21
Figure 18:	Dashboard Code.....	21
Figure 19:	Post Records	22
Figure 20:	Sign up screen page	22
Figure 21:	Model Folder	23
Figure 22:	Post list adapter folder	23
Figure 23:	Dashboard	24
Figure 24:	DesktopApplication.java part 1	24
Figure 25:	DesktopApplication.java part 2	25
Figure 26:	LoadPosts.java	25
Figure 27:	Login.java	26
Figure 28:	Controller.php	26
Figure 29:	PostHasImagesController.php	27
Figure 30:	UserController.php part 1	27

Figure 31: UserController.php part 2.....	28
Figure 32: UserController.php part 3.....	28
Figure 33: Api.php part 1.....	29
Figure 34: Api.php part 2.....	29
Figure 35: Channels.php.....	30
Figure 36: Console.php.....	30
Figure 37: Web.php	31
Figure 38: Authenticate.php	31
Figure 39: EncryptCookies.php.....	32
Figure 40: Preventrequestduringmaintenance.php	32
Figure 41: Redirectifauthenticated.php part 1	33
Figure 42: Redirectifauthenticated.php part 2	33
Figure 43: TrimStrings.php.....	34
Figure 44: TrustsHosts.php.....	34
Figure 45: Trustproxies.php part 1	35
Figure 46: trustproxies.php part 2	35
Figure 47: VerifycsrfToken.php	36
 Table 1: Methodology	10
Table 2: Issues faced, and actions taken.....	15
Table 3: Development Phases	16
Table 4: Risk Assessment.....	Error! Bookmark not defined.
Table 5: Quality Plan.....	37

Acknowledgement

First and foremost, we'd like to extend our sincere gratitude towards Mr. Pramudya Thilakaratne, our module lecturer. We are extremely humbled and grateful to have been able to receive his mentorship, guidance, and support.

The overall accomplishment of this project demanded a significant amount of guidance from many individuals. As a team, we are extremely fortunate to have had this from start to finish.

Finally, we would not have been able to successfully complete this assignment without the challenging work and assistance of all the team colleagues itself. We all enjoyed collaborating with each other.

Introduction to the Distributed System

The use of weather reporting in day-to-day life is especially important. Its utilization could greatly influence the outcome of a scenario. It could be something as simple as deciding whether you should take your umbrella on your way out, or even as major as handling cultural operations, agriculture and farming or livestock protection implementations.

With Sri Lanka's weather rapidly varying, weather can transition from largely homogeneous temperatures to torrential rain in a heartbeat. With the adaptation of unexpectedly frequent changes, it is crucial to make sure to be prepared.

Sri Lanka is liable to prevailing and predictable effects of climate change. Preceding natural disasters (such as tsunamis, floods, landslides, droughts, and cyclones) can greatly substantiate this. Profoundly weather-sensitive sectors in Sri Lanka include transport, agriculture, construction, energy, and disaster risk management.

It is important to note that even with weather stations, live broadcasts, and television/radio stations existent, not all are aware of the continuously differentiating weather conditions.

The impacts and effects of climate change could invite climate change-induced hazards and disasters. The unawareness of weather reports may affect the lives of many. Thousands of citizens are prone to being affected by the threat of climate change. They may find themselves in life threatening situations. Weather prediction is essential in order to provide citizens with pragmatic information. Furthermore, this also aids in the reduction of weather-related losses, personal safety and health, enhancement of societal benefits and in supporting economic prosperity.

It is abundantly clear that weather forecasting reports are essential to mitigate the effects civilians getting caught in a severe crisis.

Introduction to Quick Weather –

What is distributed system?

A distributed system is a collection of independent computers that appears to its users as a single coherent system. (Tanenbaum)

We have a system or application that could be a software. In that particular system we have a number of independent computers (individual computers inside the system). These individual computers are doing their own task in the system, but for the end user this looks like one single system. Those computers are doing their own tasks but for the user, they can see everything as one system.

A distributed system is contrived to assist the development process of services and applications. These services and applications are able to manoeuvre a physical architecture that would contain multiple independent processing elements. The

processing elements do not share primary memory but do partake in complying with dispatching asynchronous messages via communication network.

Furthermore, Distributed Systems also utilize independent hardware in furtherance of creating a software. It would appear as a complete system all working as a single unit, but it makes use of multiple computers in contemplation of making a distributed system function.

The primary memory of other hardware elements is not employed by Distributed Systems. Instead, multiple independent processing elements within the hardware are utilized. Using asynchronous messages, they can communicate over a networked communication.

1.1 What is Quick Weather?

In the following report, we would like to highlight the main outcome of our system. To make the reader of this report get a clear idea, we have simply built a Web Application and Mobile Application that will be running as a client application.

The public (users) often need to be informed on the current weather situation. Yet, as it is reckoned that Sri Lanka is still a developing country, it is only specified in either the news or radio stations.

In normal circumstances, the user may only be able to view the weather around their current location, but with Quick Weather, any user can view the weather around the country.

Our goal is to present a high-performance Distributed Weather System that encompasses paramount customer satisfaction, convenience, imperative security, user-friendliness, and maximum authenticity.

This system will present the needed weather for any planned activities in their personal lives. Furthermore, registered users have the ability to publish and access Blog Posts within the website. This feature will aid in giving any user a more visual idea on what the weather situation is like in that particular area. Blog Posts also anchor in many benefits, as it is a superb way to drive traffic to the website and also in building credibility.

Usefulness of the application to the real-world

Quick weather can perform 03 main functions

- Users possess the ability to yield a weather forecast at any location within the country with the utmost accuracy and efficiency of the mobile and web application. The user simply must enter the name of the location in it to access weather details.
- Registered users can upload and view posts about weather from the mobile and web application. This feature will aid in giving any user a more visual idea on what the weather situation is like in that area.
- Users can view post descriptions by using the standalone application with NetBeans.

Scope of Project

The following is a list of the application's scope:

- This program can show the state of the atmosphere for a specific place.

- By checking uploaded posts (uploaded by registered users), it is possible to take precautionary steps against catastrophic rains, winds, extreme high or low temperatures, diseases, and pests.
- Distribution Scalability.
- A high-performance system that encompasses paramount customer satisfaction, convenience, imperative security, user-friendliness, and maximum authenticity.
- A Desktop Application, Web Application and Mobile Application.

1.2 Diagrams

1.2.1 System Diagram

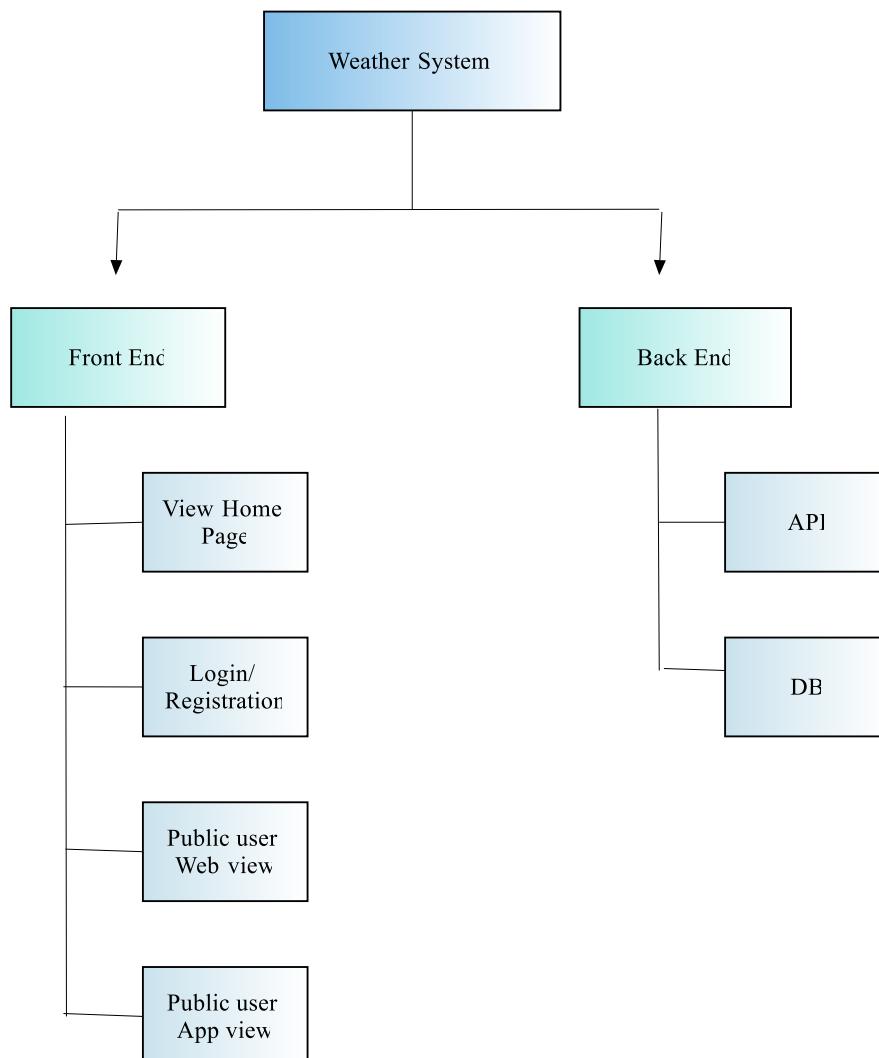


Figure 1: System Diagram

1.2.2 Context Diagram

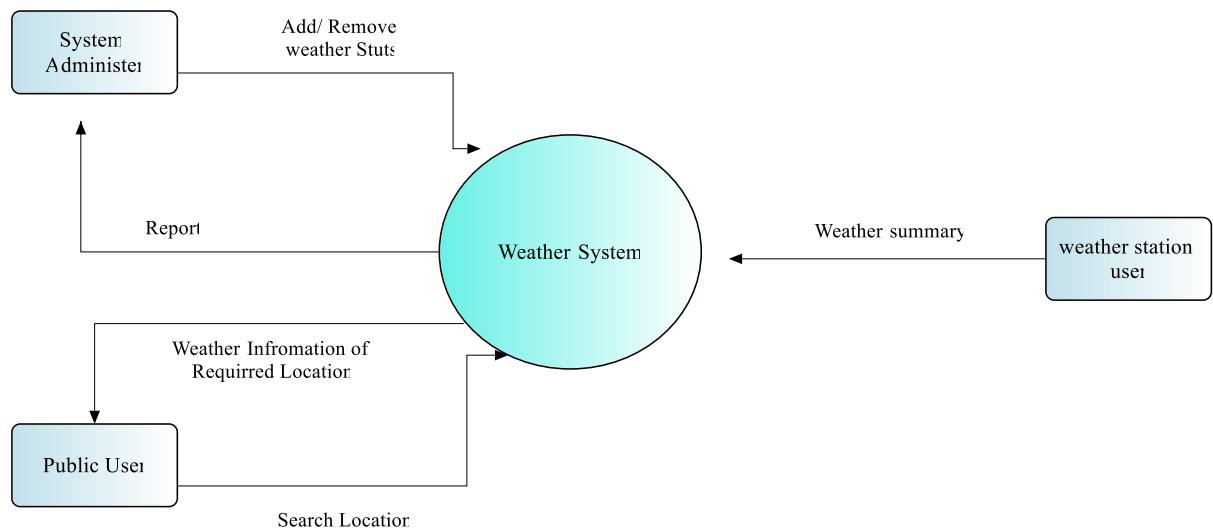


Figure 2: Context Diagram

1.2.3 Use Case Diagram

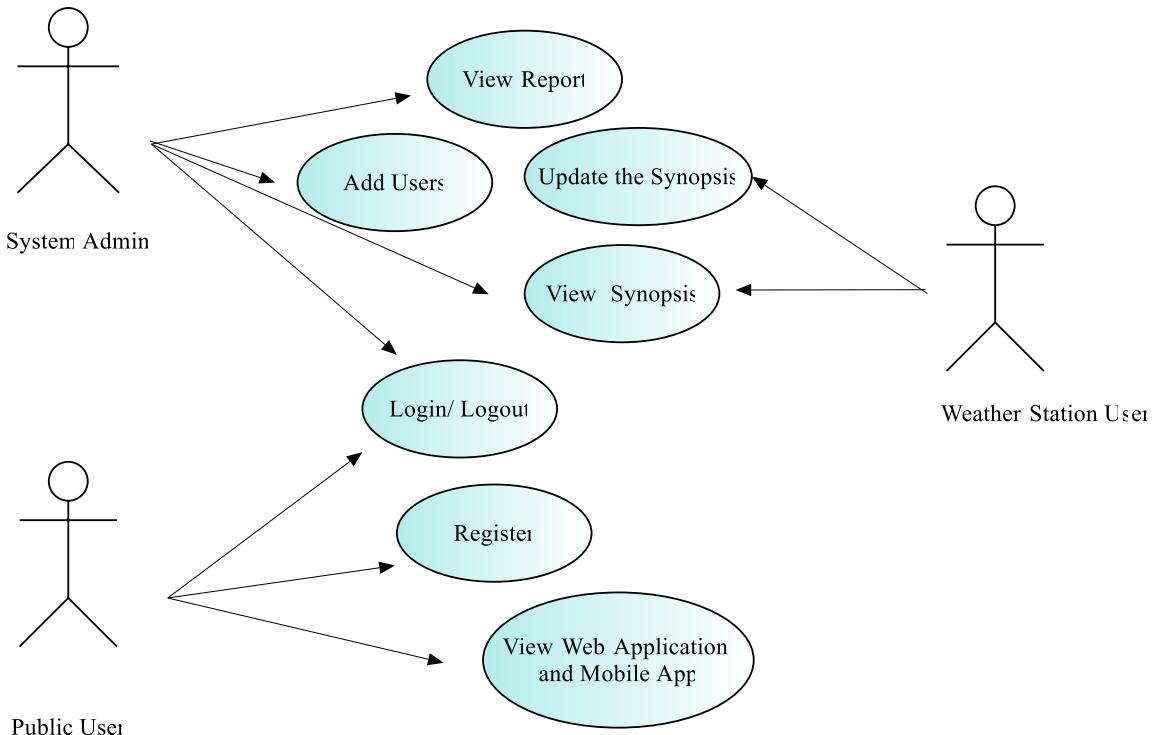


Figure 3: Use case Diagram

Mobile Application as Client

The Quick Weather mobile application is created using Android Studio, the language that Android Studio uses is Java. Since Android Studio has SDKs that run emulators, it is quick and easy to use.

The client can login/signup and post any type of image related to Technologies used:

- ✓ Laravel.
- ✓ Java in web and mobile application.
- ✓ MySQL for the database.

Methodology

Keyword	Category	Description
Java NetBeans	IDE	Utilized for the standalone Desktop Application.
PHP Laravel	Framework	Used for the API construction and Web Application Development.
PHP Laravel	Load Balancer	Distributes the web traffic amongst two or more servers and are often used for websites which receive high volumes of traffic.
Android Studio	IDE	Android studio has in built emulators which makes the system faster and more advantageous to developers.

Table 1: Methodology

Database: PhpMyAdmin MySQL

As PHP is utilized to code the Web Application, it was more coherent to utilize PhpMyAdmin for MySQL database management. PhpMyAdmin is an open-source third-party software tool, written in PHP. The primary objective of utilizing phpMyAdmin is to superintend and operate the administration of MySQL over the web. It is also feasible to run CRUD operations like databases, copy, tables, rename, databases, tables, columns, etc. phpMyAdmin can run on any server or any OS as it has a web browser. With the utilization of phpMyAdmin, it is possible to edit, create or delete the database without much difficulty. In collation to the MySQL command-line editor, it is easier to manage elements with the utilization of the phpMyAdmin graphical interface. Several servers can also be operated simultaneously. Data can also be exported into various formats like Word, PDF, SQL, XML, Spreadsheet, etc.

Gantt Chart

Time Frame			November				December				January				February				March				April				May			
Task	Progress	Week	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Initistion (Research)	100%																													
Proposal																														
Investigation and Requirement :																														
Evaluation of possible development	50%																													
Drawing sketch of system																														
New solvation and idea																														
High Level Design :																														
Designing the architecture of the system																														
Front end Development																														
Back end Development																														
System and user Acceptance Testing :																														
Unit Testing																														
System Testing																														

Figure 4: Gantt Chart (Part 1)

System and user Acceptance Testing :																													
Unit Testing																													
System Testing																													
Assemble and Completion :																													
Quality Control Testing																													
Risk Managemnt																													
Report Generation																													
Final Stage :																													
User Training																													
System Maintenance																													
Web and Mobile application release																													

Figure 5: Gantt Chart (Part 2)

Selection of Middleware

Technical Diagram

1.3 Overview of Technical Diagram

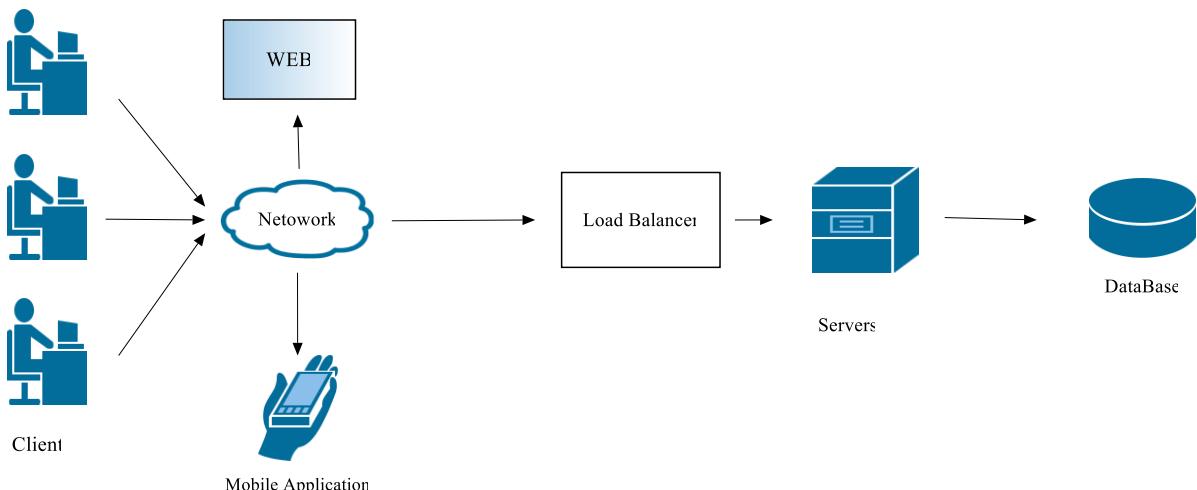


Figure 6: Overview of Technical Diagram

1.4 Technical Architectural Diagram

1.4.1 Overview of Technical Diagram

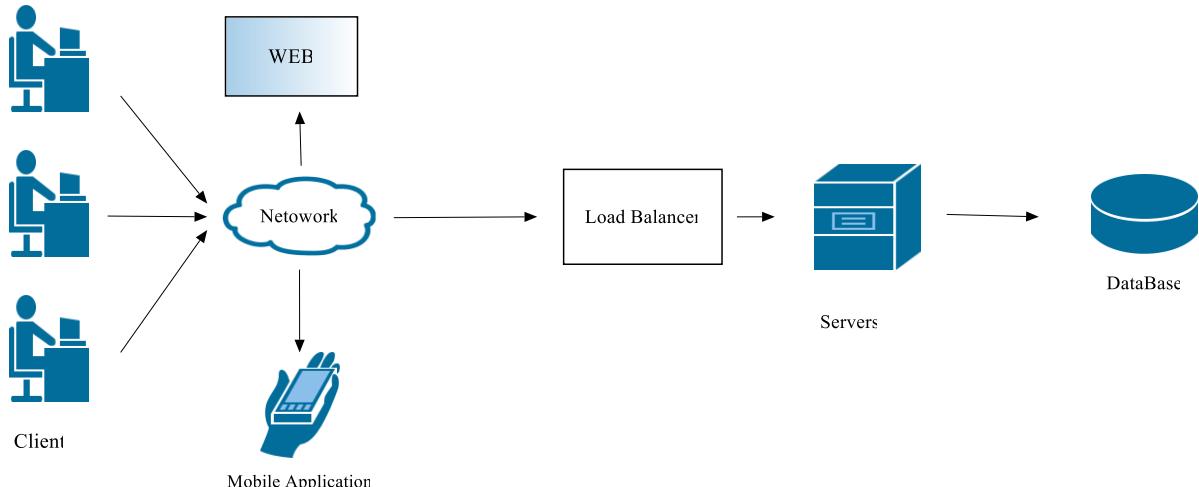


Figure 7: Overview of Technical Diagram

1.4.2 Middleware of Technical Diagram

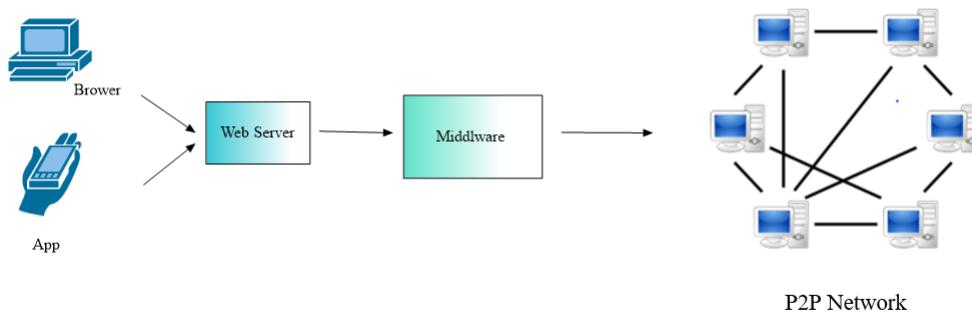


Figure 8: Middleware of Technical Diagram

1.4.3 Technical Architectural Diagram

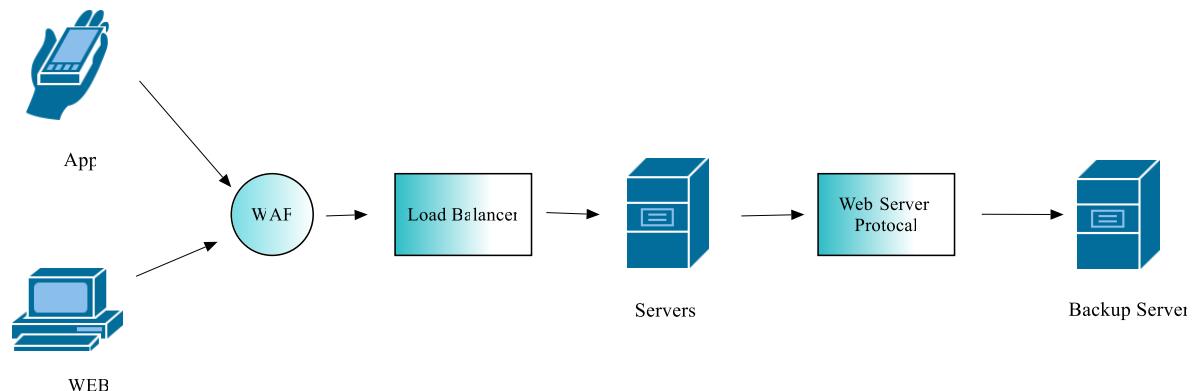


Figure 9: Technical Architectural Diagram

Our system contains a mobile app and a web app, and their total processing is made more efficient by a load balancer, which divides incoming network traffic across a cluster of three dispersed servers.

Load Balancing

A load balancer sits in front of your servers, acting as a "traffic cop," directing client requests across all servers capable of satisfying those requests in a way that maximizes performance and reformulation testing while ensuring that no single server is overworked, potentially degrading performance. The load balancer transfers traffic to the remaining online servers if a single server goes down. When a new server is added to a server group, the load balancer begins sending requests to it automatically.

A load balancer accomplishes the following tasks in this manner:

- Client requests or network load are efficiently distributed among numerous servers.
- Sends requests exclusively to online servers, ensuring high availability and reliability.
- Provides the flexibility to add or remove servers as need dictates.

1.5 With Load Balancing

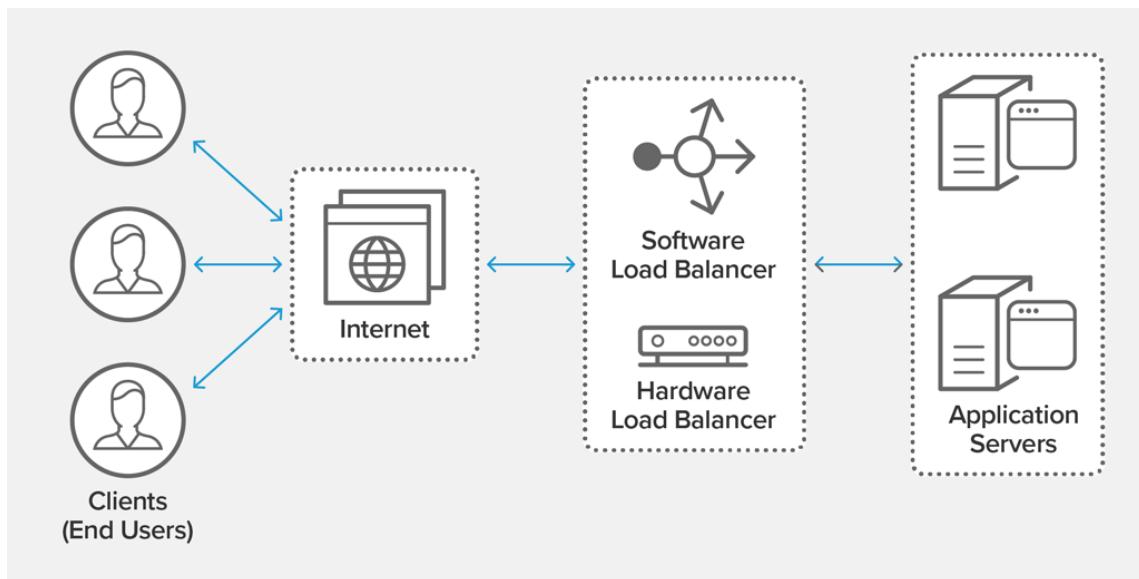


Figure 10: With Load Balancing

1.6 Without Load Balancing

No LOAD BALANCING

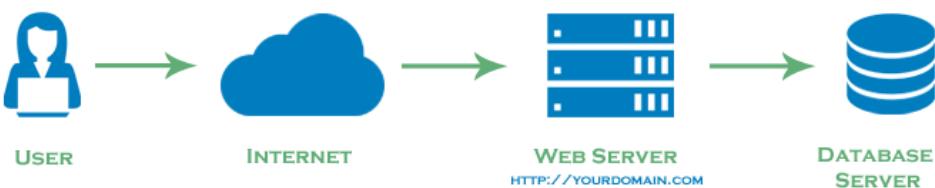


Figure 11: Without Load Balance

Tolerance to Network Failure

As our system runs in three servers beside the load balancer, just in case of a network failure it automatically switches to the backup server, thus no network failures can influence the system function.

System Architectural Diagram

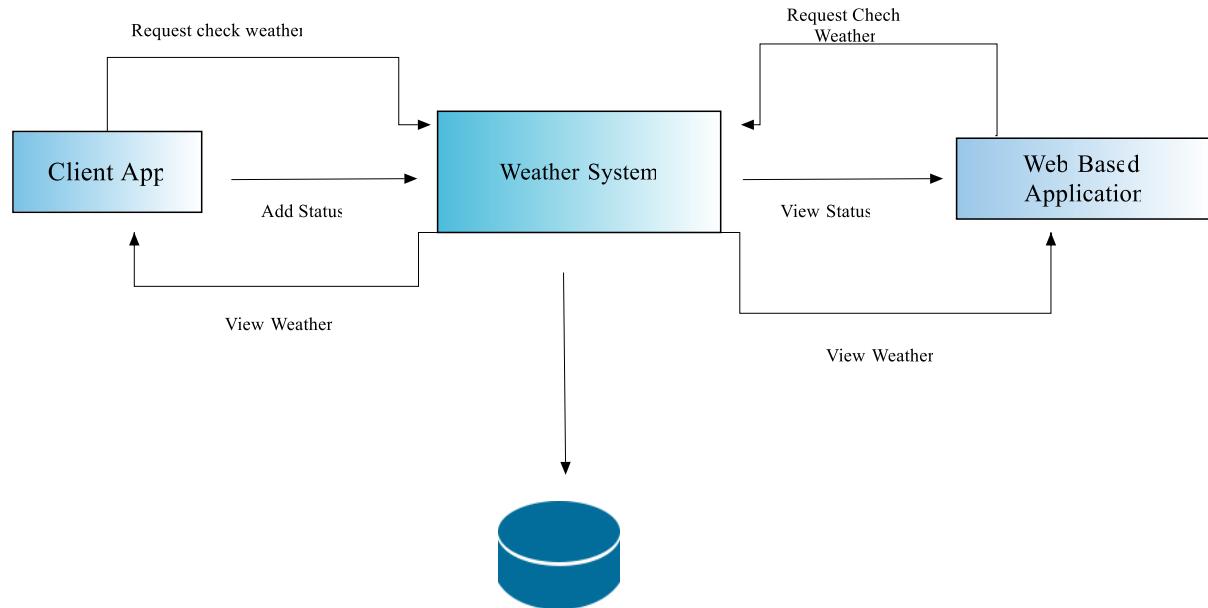


Figure 12: System Architectural Diagram

1.7 User Requirements

The Graphical User Interface system is designed in a way where the user can:

- Users can check weather without having to sign up/sign into the system.
- The users just must enter the city name of the preferred place where they want to know the weather of.
- The weather will be displayed in an instance for the user to view.
- Users also can check posts that are posted by other members of society.

1.8 Functional Requirements –

- We tend to run multiple copies of each element on different machines.
- Data is replicated in order to avoid single machine failure. In this way, loss of data can be avoided.
- The system runs elements on separate machines.
- Has redundancy at all layers.

API / Middleware Justification

Initially when we run the system, we have used middleware as the “glue” holding the applications made together, with this “glue” we can connect the three applications together.

For the implementation of the middleware Laravel was used.

Functions done by the middleware:

- Authenticates the user.
- Executes code.

Software components used

- Laravel – Laravel was used to create the API for the system, this is a free, open-source web framework that is used for web application development. It follows a model-view-controller design pattern.
- MySQL – Open-source relational database management system, it uses Structured Query Language, one of the most famous languages for managing content in a database.
- Android Studio – Android Studio is a mobile application development platform; Android Studio has their own emulators which makes it easy for developers to use and maintain. Android studio has fast execution, which provides time effectiveness for developers.
- NetBeans – integrated development environment for Java. NetBeans was used to create our stand-alone application.

Issues faced and actions taken

Issues Faced	Actions Taken
Software Installation Failure	With necessary research by following tutorial and LinkedIn courses, we manage to resolve the issues that had materialized.
Connecting to API	
“PHP artisan migrate” did not migrate	

Table 2: Issues faced, and actions taken

Development phases

Development phases	Description
1. Planning	When it comes to defining a project's boundaries, it is mandatory to consider the potentiality of how large the scope can be. By composing a project plan, a calculation of the spatial sense (timeliness) of the project will be incited. This will include a breakdown of tasks, a conclusion of the inherent deadlines and overall time taken to complete the tasks. All necessary requirements are accumulated to determine the sequencing the work, including a creation of an initial project schedule.
2. Analysis	The analysis stage will decipher the system's critical features and requirements. This will establish whether it is worthwhile to develop. It will also determine a requirement of the possible tools and technologies to be utilized, a potential project schedule, necessary budget estimations, etc.
3. Designing	Designing will aid in roughly determining the shape and form of what must be assembled. It will be utilized in the form of a template. This will dictate the implementation proceedings, software architecture, database, and frontend designs.

4. Coding and debugging	Frontend development, Database development, Website creation, and the implementation and development of backend logical coding has been contrived. Correspondingly, a Mobile Application with implemented logical coding and a frontend is generated. In addition, an API to interface and a MySQL database has been created.
5. Testing	Functional testing is executed to see if we are achieving the expected and needed outcomes. All code units in the project have been tested, closely following the unit testing process. Integration Testing has been conducted after Once the system's modules were integrated, the full system was effectuated with end-to-end testing. We have also exercised User Interface Testing. This process would entail checking whether the UIs have been designed in fulfilment of the system's specifications. Furthermore, Load Testing has been conducted within the confines of non-functional testing to emphasise the system's overall quality.
6. Deliver the product	In the final development stage, the website, mobile application, and report is completed and presented.

Table 3: Development Phases

Tasks undertaken

1.9 Database

Shown below are the Database tables which contain important info, such as storing users' data and other information, post information and city data.

Table	Action	Rows	Type	Collation	Size	Overhead
city_data		0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
failed_jobs		0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
migrations		6	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
password_resets		0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
personal_access_tokens		0	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-
posts		2	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
users		2	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
7 tables	Sum	10	InnoDB	utf8mb4_general_ci	192.0 KiB	0 B

Figure 13: Database

User's data is stored as follows.

					remember_token	created_at	updated_at
<input type="checkbox"/>		1	test	test@gmail.com	NULL	\$2y\$10\$8os5MQHfKec.zTEqhiHTu9yG89lPBETail2wbp3AOg...	NULL
<input type="checkbox"/>		2	Niihan	neo@gmail.com	NULL	\$2y\$10\$Dfex38ZjwhoU5q3AQ4qKe9WvkcWY2dsd4melniipA...	NULL

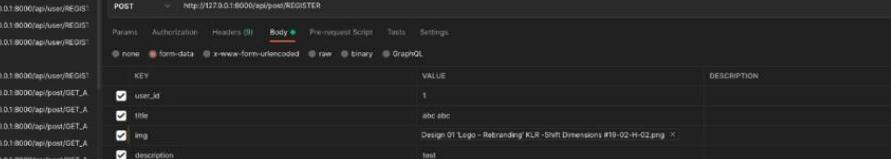
Figure 14: Stored data of users

Posts submitted by users are stored in here.

					description	status	created_at	updated_at
<input type="checkbox"/>		1	1	test post	test post desc	1	NULL	NULL
<input type="checkbox"/>		2	1	test title	1652032994.png	data	2022-05-08 18:03:14	2022-05-08 18:03:14

Figure 15: Post submission

1.10 Postman API requests and testing



The screenshot shows the Postman application interface. On the left, a sidebar displays 'Collections' (including 'Today' and 'Yesterday') and 'Recent' (including 'Mock Servers', 'Monitors', 'Flows', and 'History'). The main area shows a collection of API requests. A specific POST request for 'REGISTER' is selected, showing its details. The 'Body' tab is active, displaying the following JSON payload:

```
POST http://127.0.0.1:8000/api/post/REGISTER
{
  "user_id": 1,
  "title": "abc abc",
  "img": "Design 01 Logo - Rebranding KLR-Shift Dimensions #10-02-H-C2.png",
  "description": "test"
}
```

The 'Body' tab also includes a 'Pre-request Script' and 'Tests' section. The 'Headers' tab shows the following headers:

Header	Value
Content-Type	application/json

The 'Test' tab contains the following assertions:

```
1. code: 200
2. "msg": "Successfully Saved Post"
3. "status": "Success"
4. "data": null
```

At the bottom, the status bar shows 'Status 200 OK' and 'Time: 27 ms'. The 'Save Response' button is also visible.

1.10.1 Add Post Request



http://127.0.0.1:8000/api/user/REGISTRATION?name=Niran&email=niran@gmail.com&password=Niran@123&password_confirmation=Niran@123

POST http://127.0.0.1:8000/api/user/REGISTRATION

Params: Authorization, Headers (8), Body, Pre-request Script, Tests, Settings

Query Params

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> name	Niran	
<input checked="" type="checkbox"/> email	niran@gmail.com	
<input checked="" type="checkbox"/> password	Niran@123	
<input checked="" type="checkbox"/> password_confirmation	Niran@123	

Body: `{}
{"code": "200", "message": "Success", "data": { "id": 3 }}`

Body: Cookies: Headers (10): Test Results: Status: 200 OK Time: 160 ms Size: 490 B Save Response:   

1.10.2 Login Request

1.11 Screenshots of the website

The web application as shown below is the interface the user is greeted with when

Weather Posts

Weather Details

Login

colombo

City	Temprature	Weather	Wind Speed
Colombo	29°	Rainy	9.3 knots

Weather Posts

Weather Details

Login

Hello, world!

This application shows the posted weather informations, please use below search bar for search post by title.

Search by post title

Search Reset

test1

test desc

Add Post

x

Add Post Title

Description

Post Image

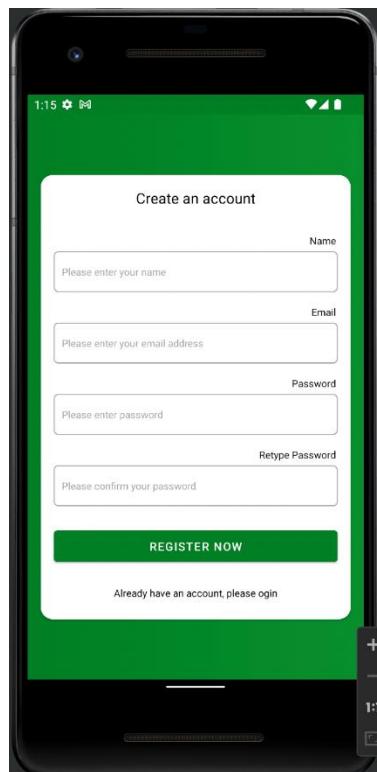
Choose file No file chosen

Close

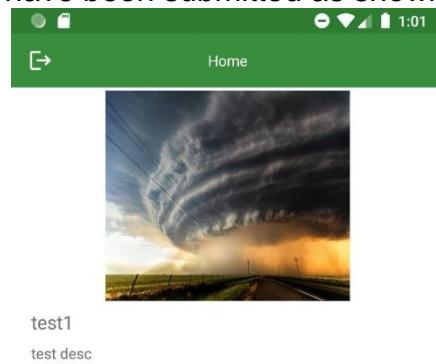
Submit

1.12 Screenshots of the Mobile Application

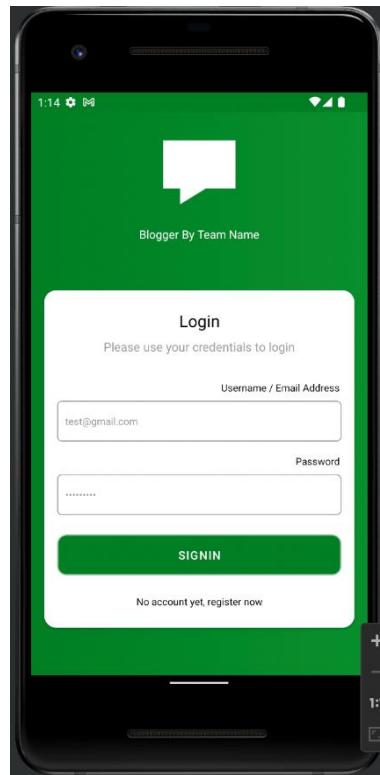
This is the interface the client is greeted with when he/ she is in need of creating an account to gain privileges of posting updates on the weather post page on the web page of this distributed application.



After the user creates an account, he or she can login through the app to see the weather-related posts that have been submitted as shown below



This is the user login page where he or she enters their email and password to login to the app to view the posts.



1.13 Main codes of the website

1.14 Main codes of the mobile application

Main activity is where the code responsible for the temporary splash screen is visible as you access the mobile application.

1.14.1 Create an account mobile application code

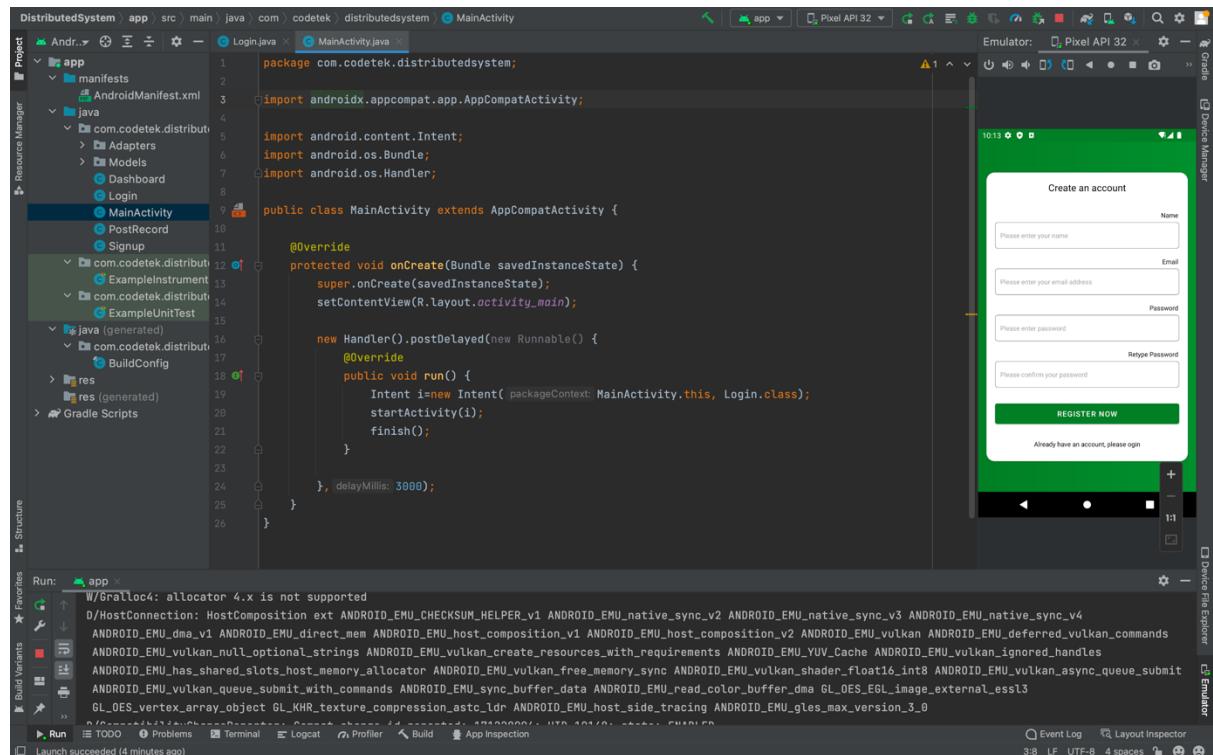
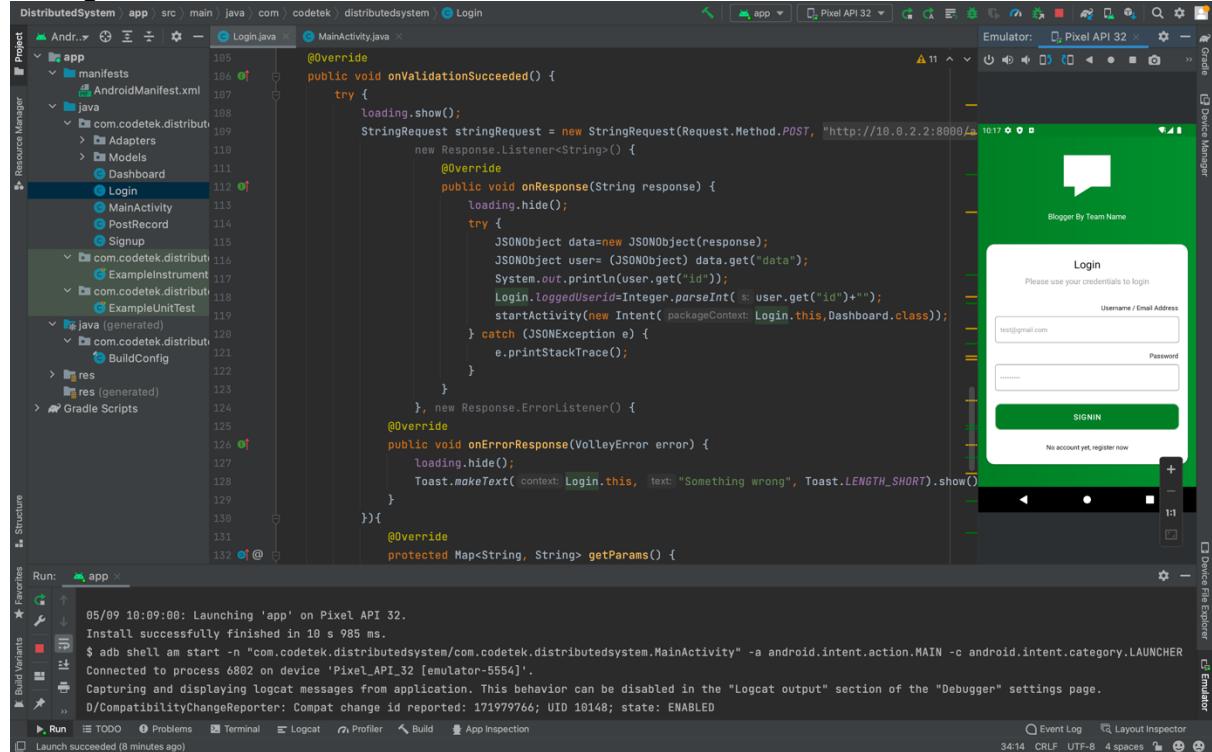


Figure 16: Create an account

1.14.2 Login

Login screen code is present in this code view as shown with the running interface in the right side of the screen shot.



The screenshot shows the Android Studio interface with the project structure on the left and the code editor on the right. The code editor displays the `Login.java` file, which contains the logic for handling user login. The running interface on the right is a mobile login screen with a green header and a white form. The form has fields for 'Username / Email Address' and 'Password', and a 'SIGNIN' button. Below the button is a link 'No account yet, register now'.

```
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
587
588
589
589
590
591
592
593
594
595
596
597
597
598
599
599
600
601
602
603
603
604
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1543
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1550
1551
1551
1552
1552
1553
1553
1554
1554
1555
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1560
1561
1561
1562
1562
1563
1563
1564
1564
1565
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1570
1571
1571

```

1.14.4 Post records

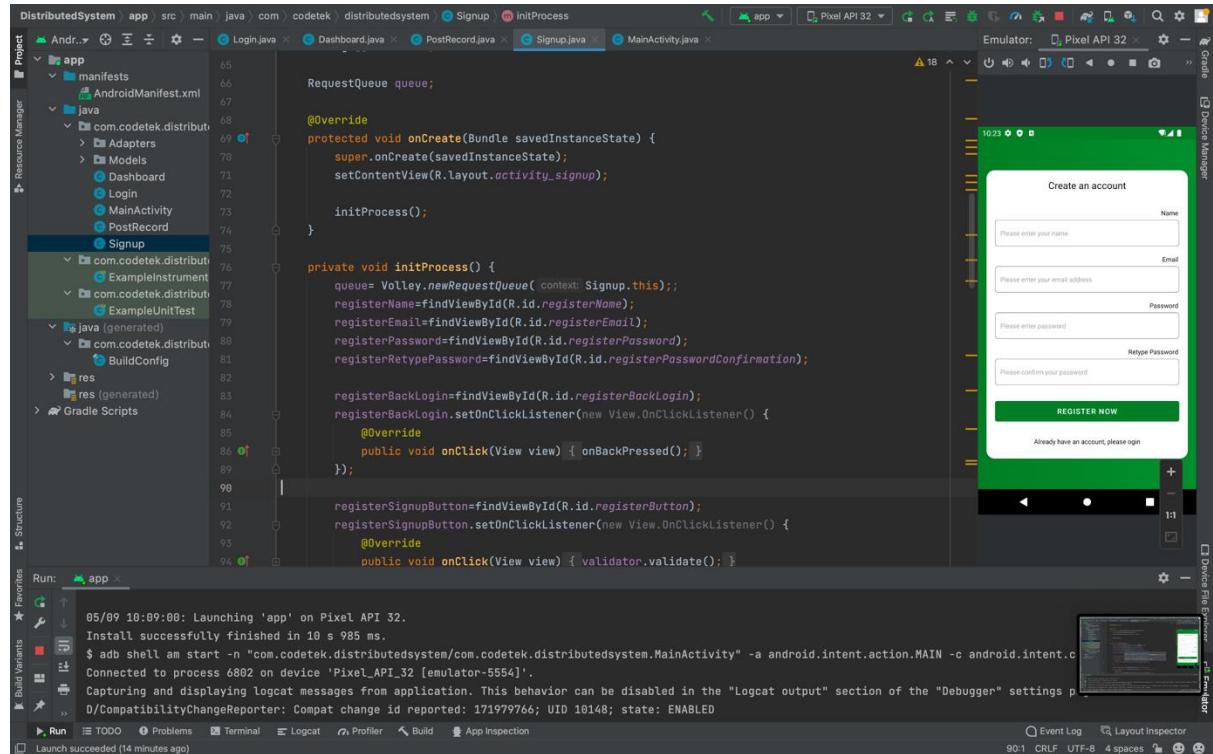


Figure 19: Post Records

1.14.5 Signup Screen

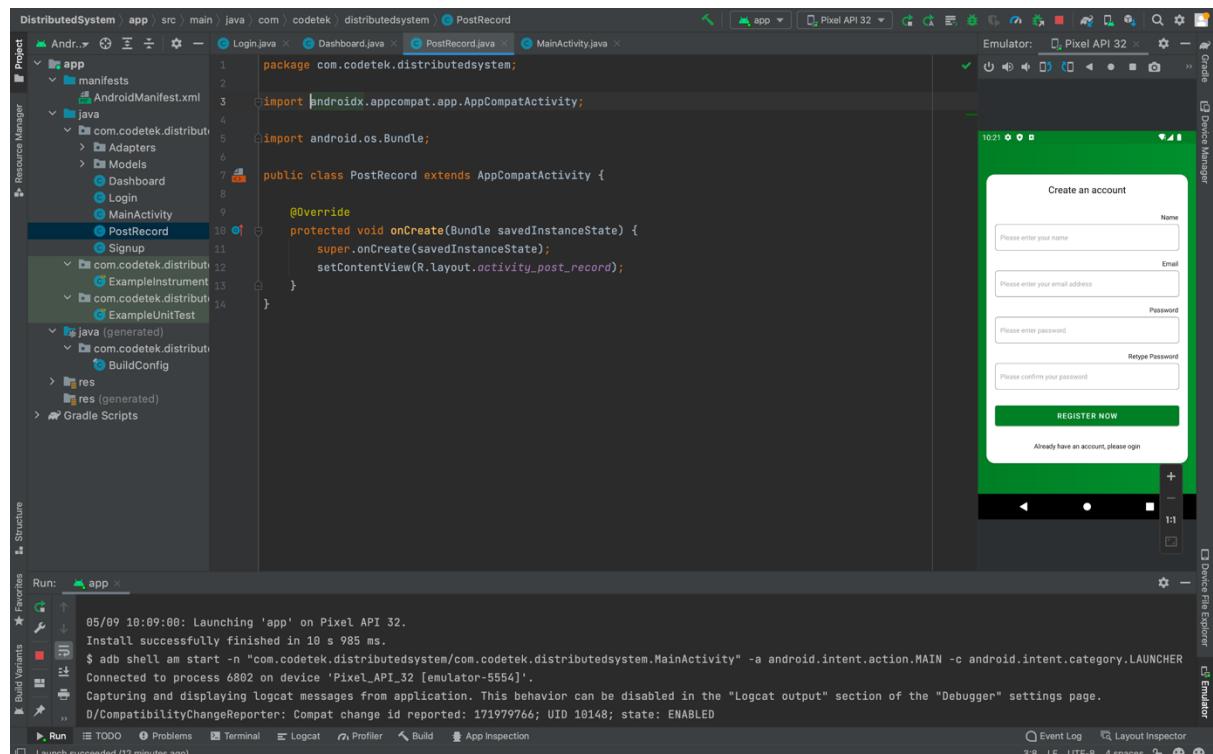
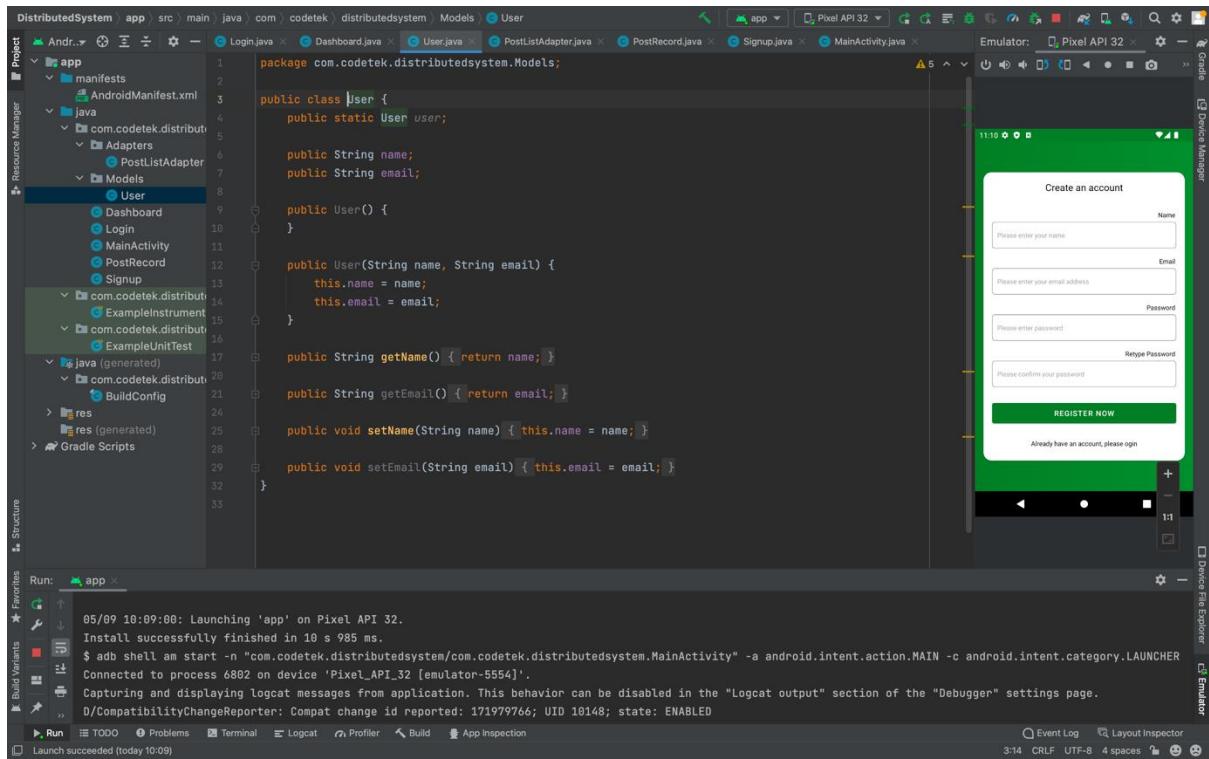


Figure 20: Sign up screen page

1.14.6 Models Folder

The model folder is responsible for storing user data and serialization of that data.



```

package com.codetek.distributedsystem.Models;

public class User {
    public static User user;
    public String name;
    public String email;

    public User() {
    }

    public User(String name, String email) {
        this.name = name;
        this.email = email;
    }

    public String getName() { return name; }

    public String getEmail() { return email; }

    public void setName(String name) { this.name = name; }

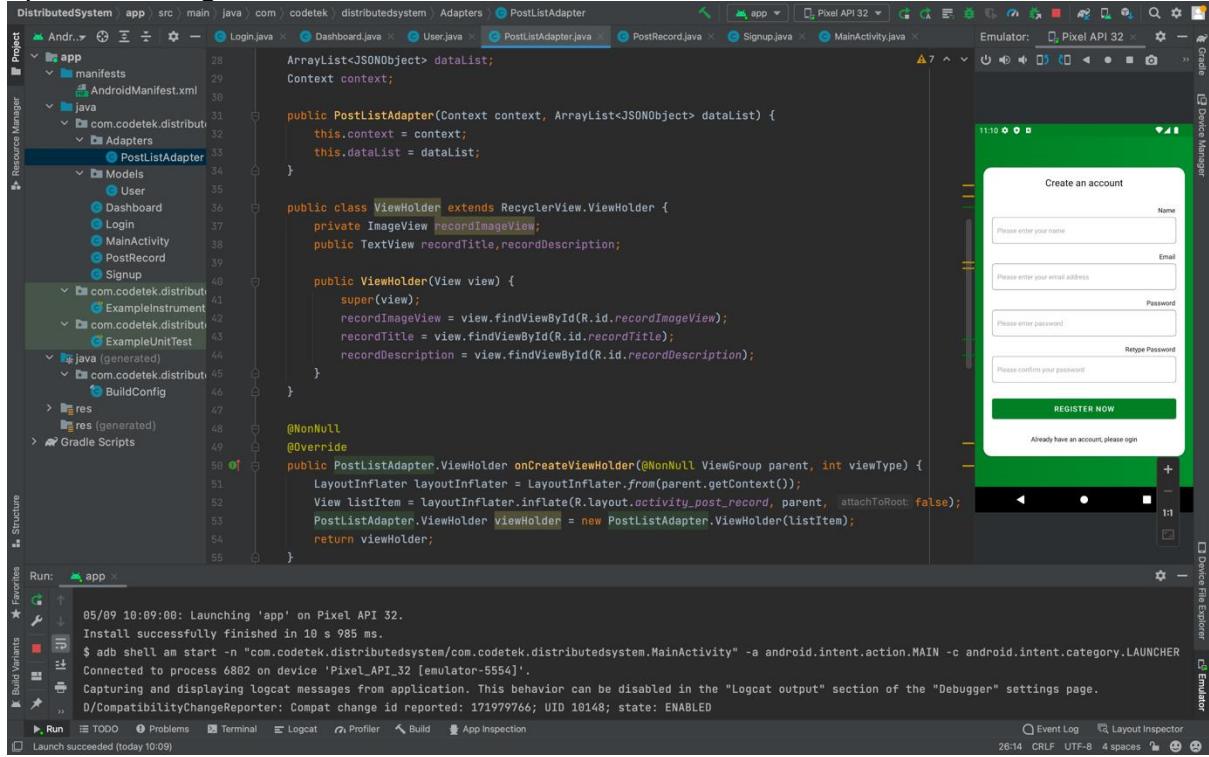
    public void setEmail(String email) { this.email = email; }
}

```

Figure 21: Model Folder

1.14.7 Post list adapter folder

This block of code is responsible for the way on which the posts are displayed for the users of the application by accordingly segregating the title of post, description, and uploaded images etc.



```

ArrayList<JSONObject> dataList;
Context context;

public PostListAdapter(Context context, ArrayList<JSONObject> dataList) {
    this.context = context;
    this.dataList = dataList;
}

public class ViewHolder extends RecyclerView.ViewHolder {
    private ImageView recordImageView;
    public TextView recordTitle, recordDescription;

    public ViewHolder(View view) {
        super(view);
        recordImageView = view.findViewById(R.id.recordImageView);
        recordTitle = view.findViewById(R.id.recordTitle);
        recordDescription = view.findViewById(R.id.recordDescription);
    }
}

@NonNull
@Override
public PostListAdapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    LayoutInflator layoutInflater = LayoutInflator.from(parent.getContext());
    View listItem = layoutInflater.inflate(R.layout.activity_post_record, parent, false);
    PostListAdapter.ViewHolder viewHolder = new PostListAdapter.ViewHolder(listItem);
    return viewHolder;
}

```

Figure 22: Post list adapter folder

1.15 Main codes of the Desktop application

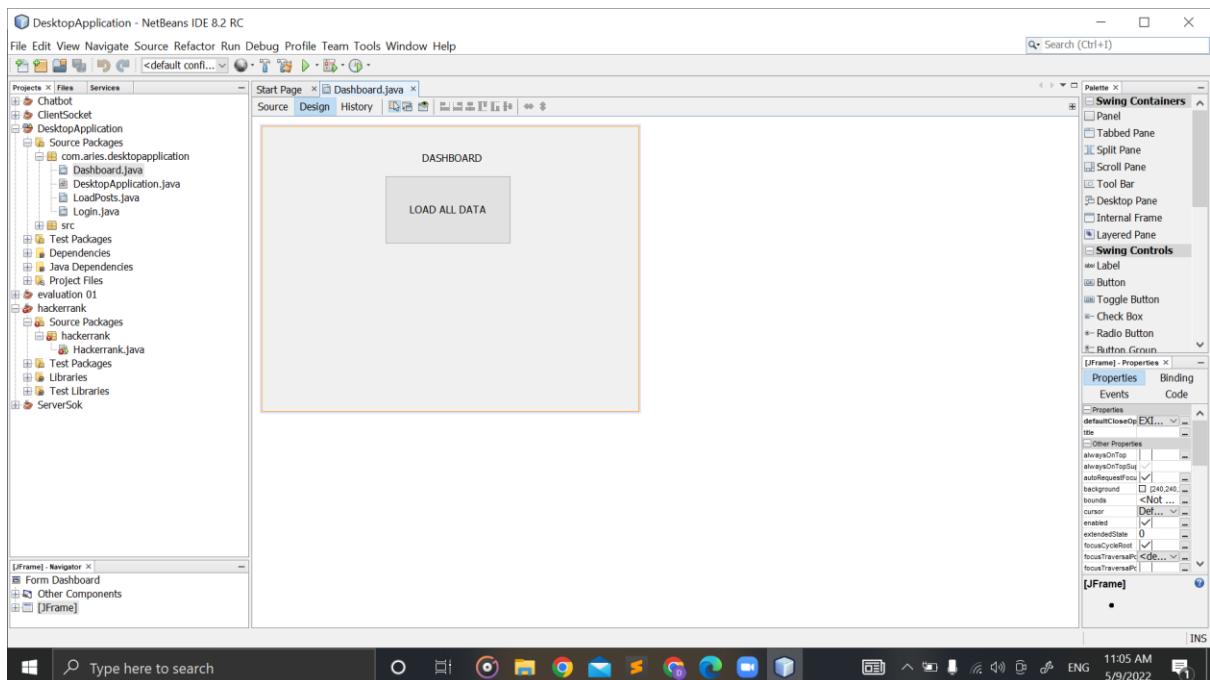


Figure 23: Dashboard

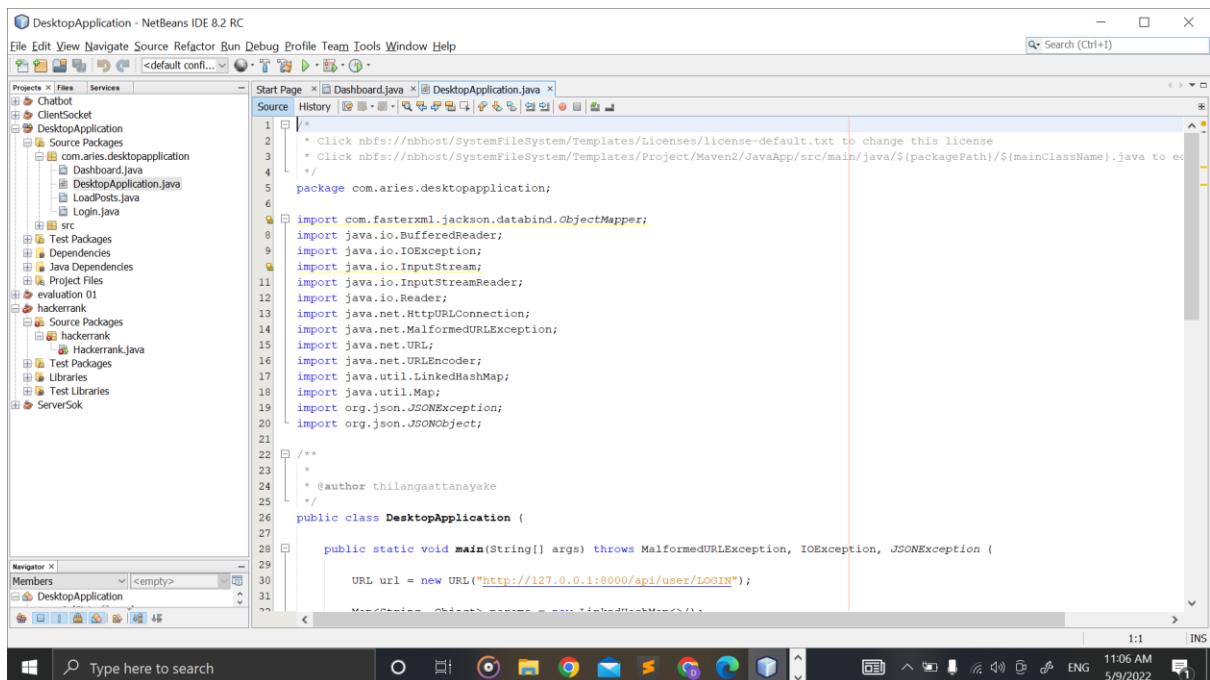
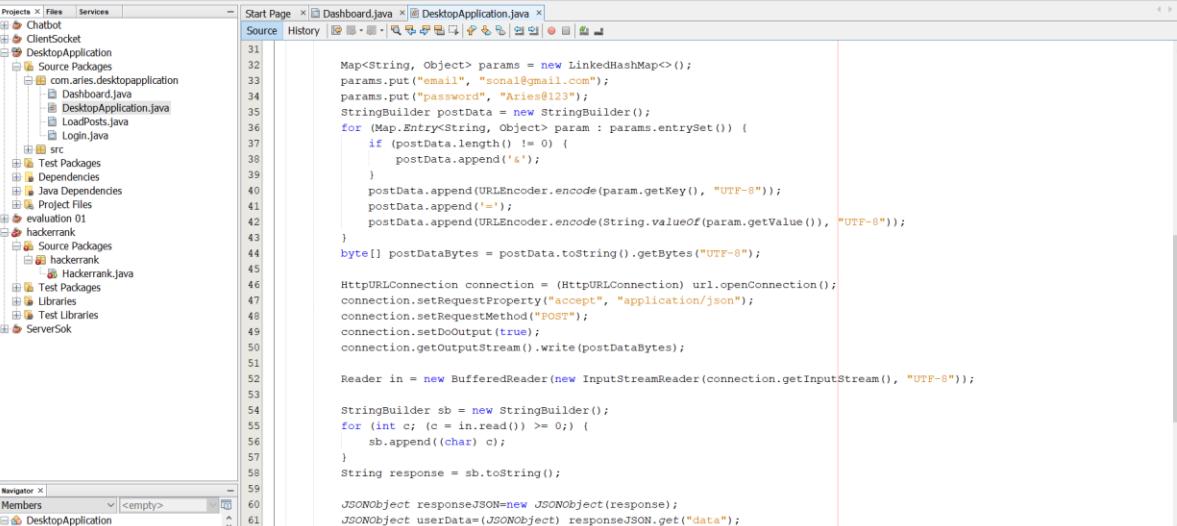


Figure 24: DesktopApplication.java part 1



DesktopApplication - NetBeans IDE 8.2 RC

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+F)

Projects default config Services

Start Page Dashboard.java DesktopApplication.java

Source History 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 <span

Figure 25: DesktopApplication.java part 2

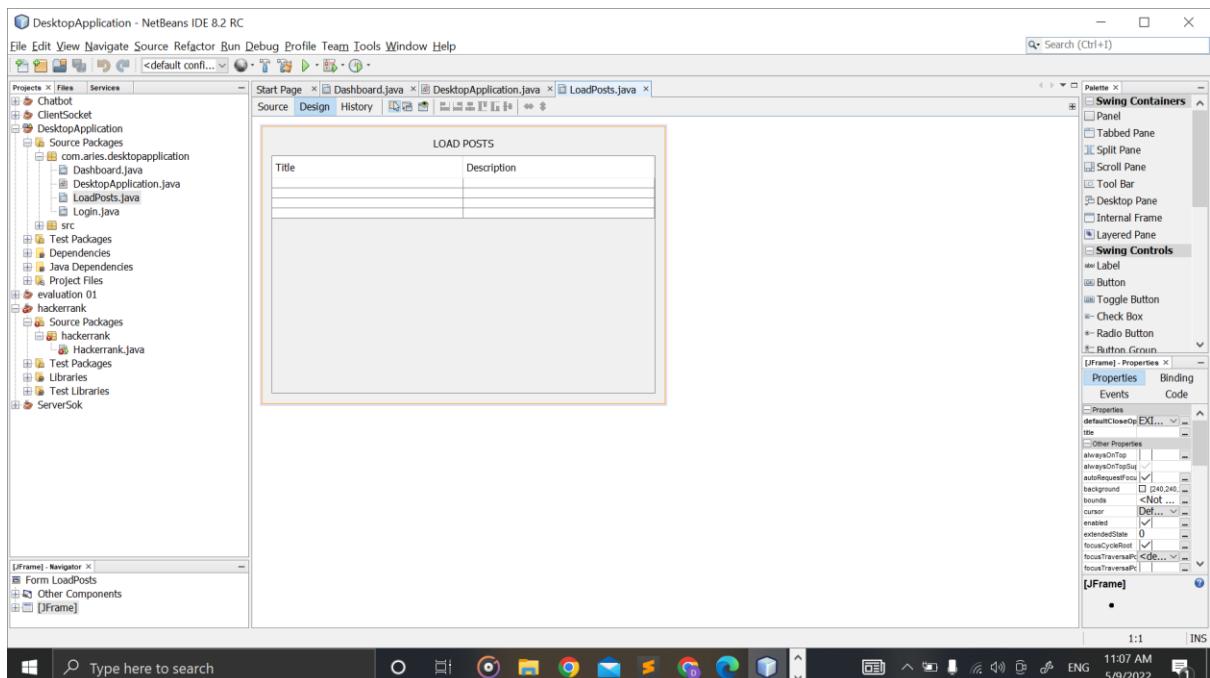


Figure 26: LoadPosts.java

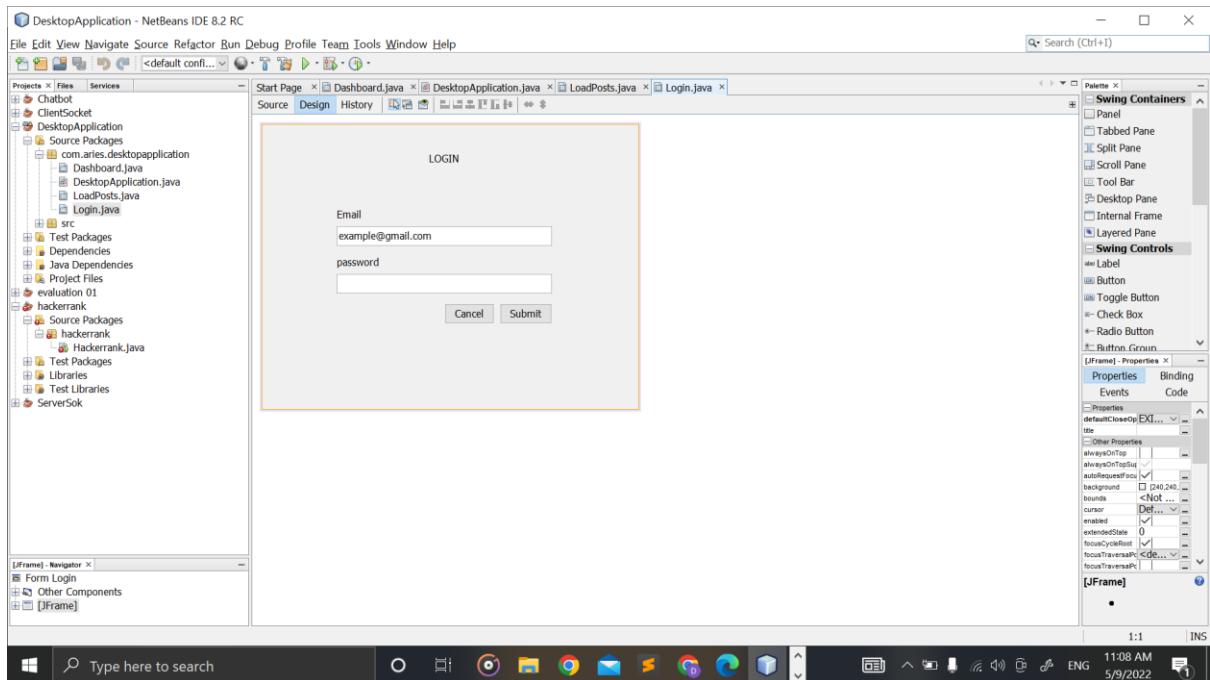


Figure 27: Login.java

1.16 Main codes of the API

1.16.1 Controller

```

Controller.php - LARAVEL-DISTRIBUTED-SYSTEM-2022 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
Controller.php - Controller.php
app > Http > Controllers > Controller.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
6  use Illuminate\Foundation\Bus\DispatchesJobs;
7  use Illuminate\Foundation\Validation\ValidatesRequests;
8  use Illuminate\Routing\Controller as BaseController;
9
10 class Controller extends BaseController
11 {
12     use AuthorizesRequests, DispatchesJobs, ValidatesRequests;
13 }

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

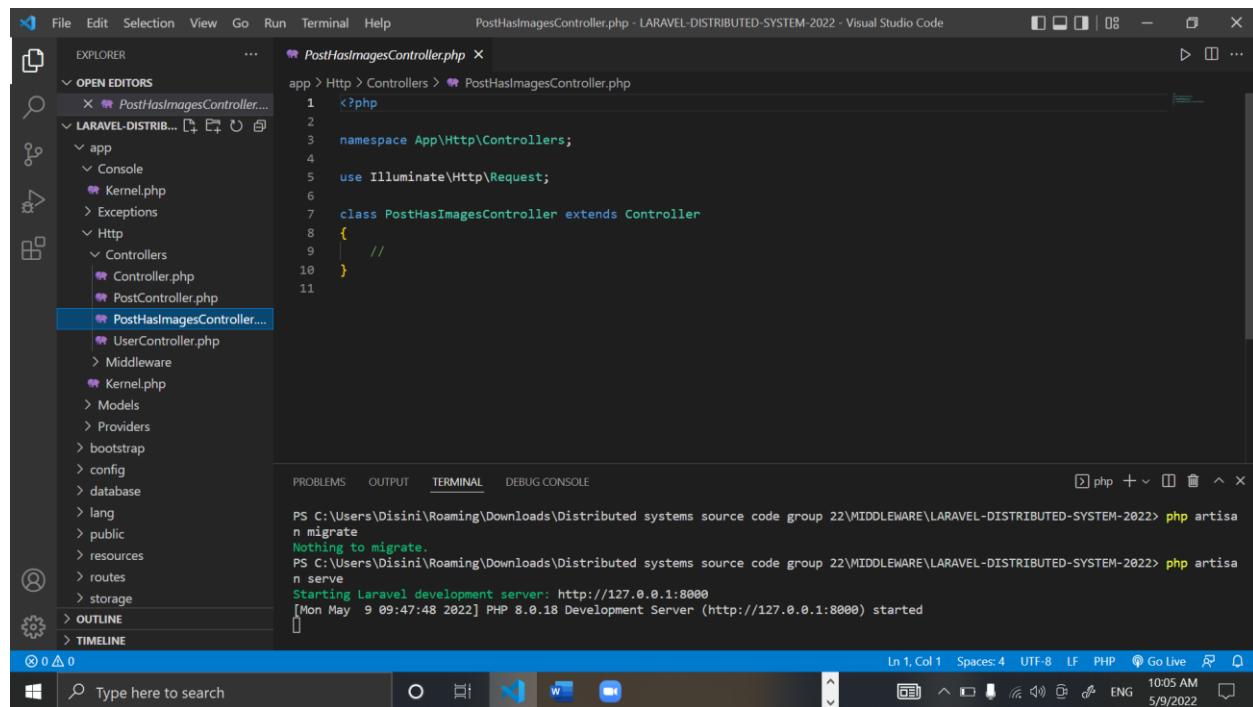
```

PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan
n migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisa
n serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started

```

11, Col 1 Spaces: 4 UTF-8 LF PHP Go Live RP Q

Figure 28: Controller.php



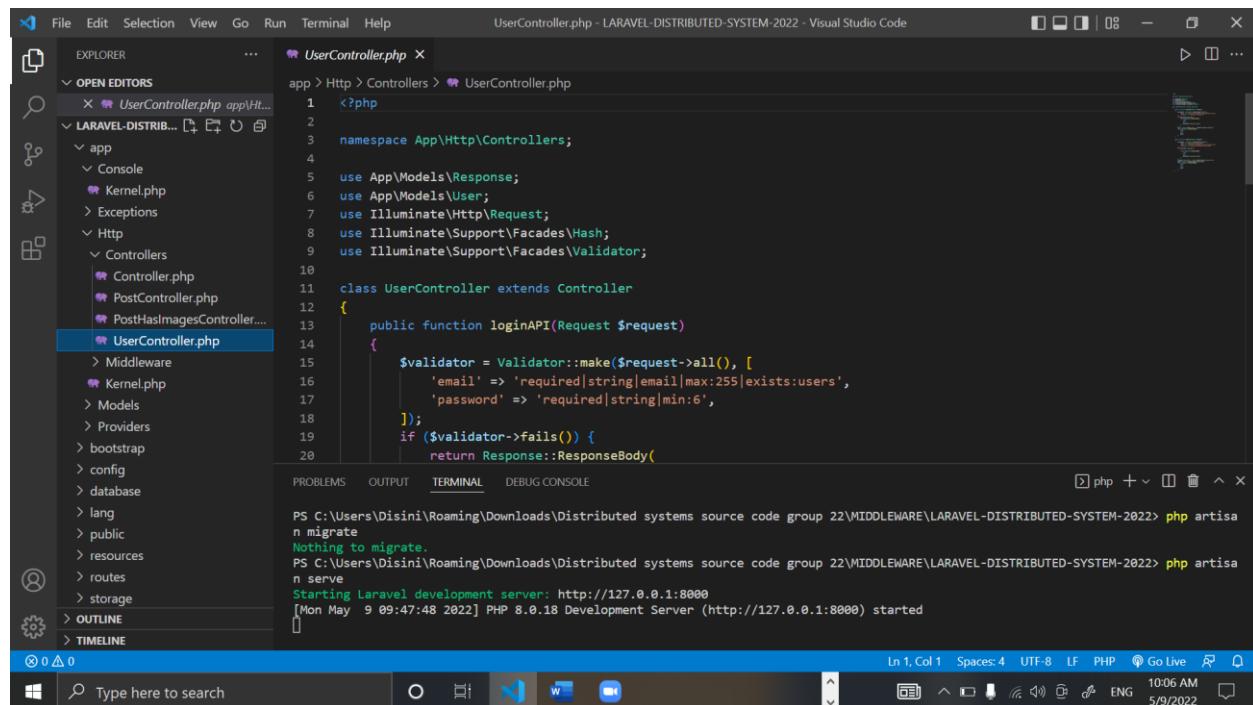
```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
class PostHasImagesController extends Controller
{
    //
}
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started
```

Ln 1, Col 1 Spaces:4 UTF-8 LF PHP ENG 10:05 AM 5/9/2022

Figure 29: PostHasImagesController.php



```
<?php
namespace App\Http\Controllers;
use App\Models\Response;
use App\Models\User;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;
class UserController extends Controller
{
    public function loginAPI(Request $request)
    {
        $validator = Validator::make($request->all(), [
            'email' => 'required|string|email|max:255|exists:users',
            'password' => 'required|string|min:6',
        ]);
        if ($validator->fails()) {
            return Response::ResponseBody(

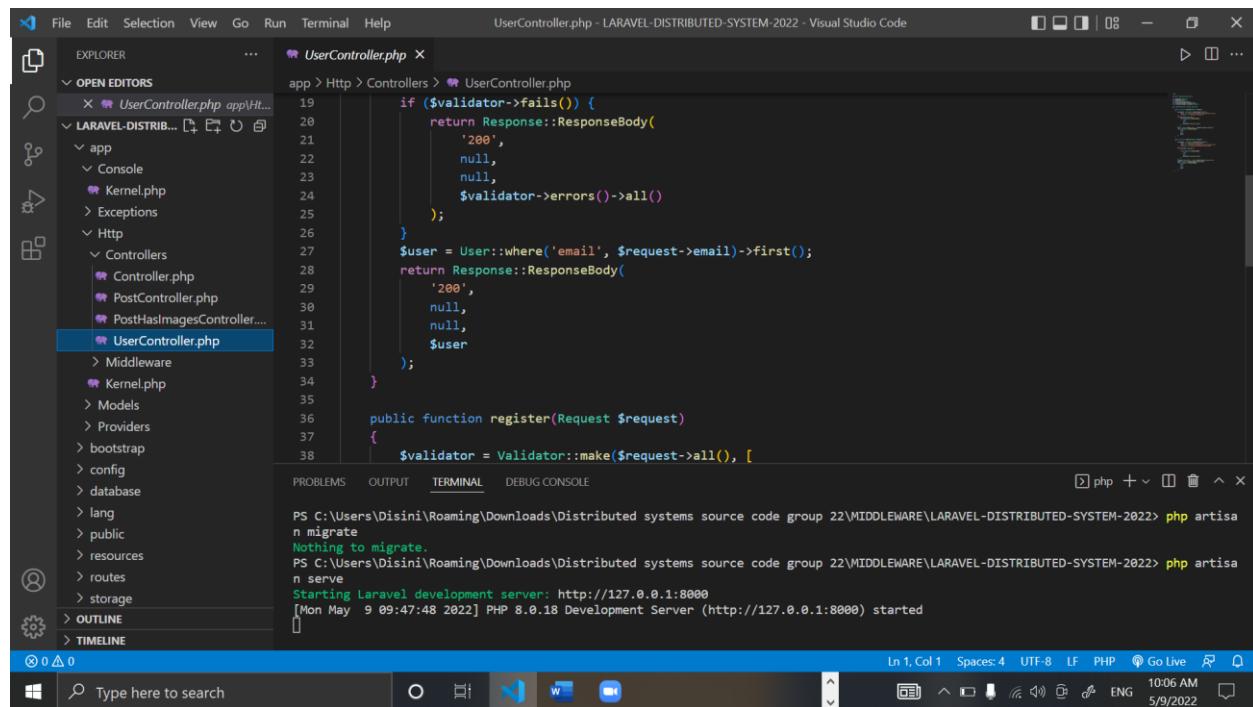
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started
```

Ln 1, Col 1 Spaces:4 UTF-8 LF PHP ENG 10:06 AM 5/9/2022

Figure 30: UserController.php part 1



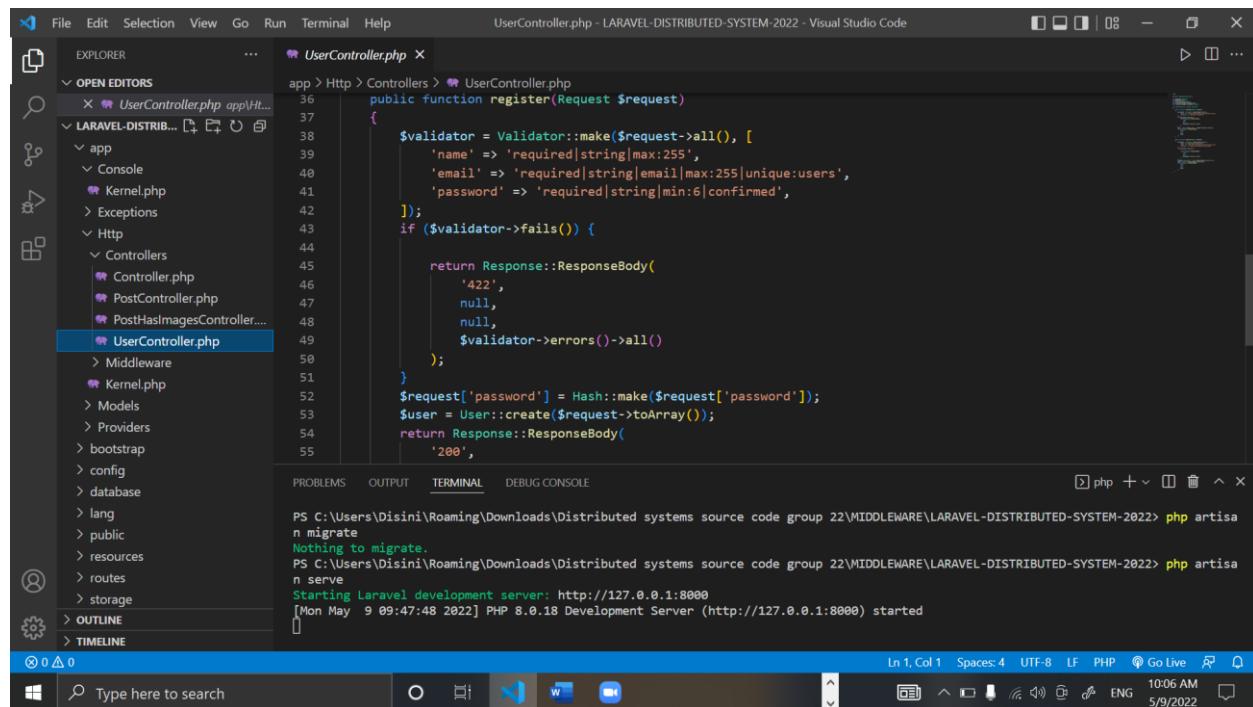
```
UserController.php - LARAVEL-DISTRIBUTED-SYSTEM-2022 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER OPEN EDITORS
LARAVEL-DISTRIBUTED-SYSTEM-2022 app Http Controllers
UserController.php
app > Http > Controllers > UserController.php
19     if ($validator->fails()) {
20         return Response::ResponseBody(
21             '200',
22             null,
23             null,
24             $validator->errors()->all()
25         );
26     }
27     $user = User::where('email', $request->email)->first();
28     return Response::ResponseBody(
29         '200',
30         null,
31         null,
32         $user
33     );
34 }
35
36     public function register(Request $request)
37     {
38         $validator = Validator::make($request->all(), [
39             'name' => 'required|string|max:255',
40             'email' => 'required|string|email|max:255|unique:users',
41             'password' => 'required|string|min:6|confirmed',
42         ]);
43         if ($validator->fails()) {
44             return Response::ResponseBody(
45                 '422',
46                 null,
47                 null,
48                 $validator->errors()->all()
49             );
50         }
51         $request['password'] = Hash::make($request['password']);
52         $user = User::create($request->toArray());
53         return Response::ResponseBody(
54             '200',
55         );
56     }
57 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\ MIDDLEWARE\ LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\ MIDDLEWARE\ LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started

Ln 1, Col 1 Spaces: 4 UTF-8 LF PHP ENG 10:06 AM 5/9/2022

Figure 31: UserController.php part 2



```
UserController.php - LARAVEL-DISTRIBUTED-SYSTEM-2022 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER OPEN EDITORS
LARAVEL-DISTRIBUTED-SYSTEM-2022 app Http Controllers
UserController.php
app > Http > Controllers > UserController.php
36     public function register(Request $request)
37     {
38         $validator = Validator::make($request->all(), [
39             'name' => 'required|string|max:255',
40             'email' => 'required|string|email|max:255|unique:users',
41             'password' => 'required|string|min:6|confirmed',
42         ]);
43         if ($validator->fails()) {
44             return Response::ResponseBody(
45                 '422',
46                 null,
47                 null,
48                 $validator->errors()->all()
49             );
50         }
51         $request['password'] = Hash::make($request['password']);
52         $user = User::create($request->toArray());
53         return Response::ResponseBody(
54             '200',
55         );
56     }
57 }
```

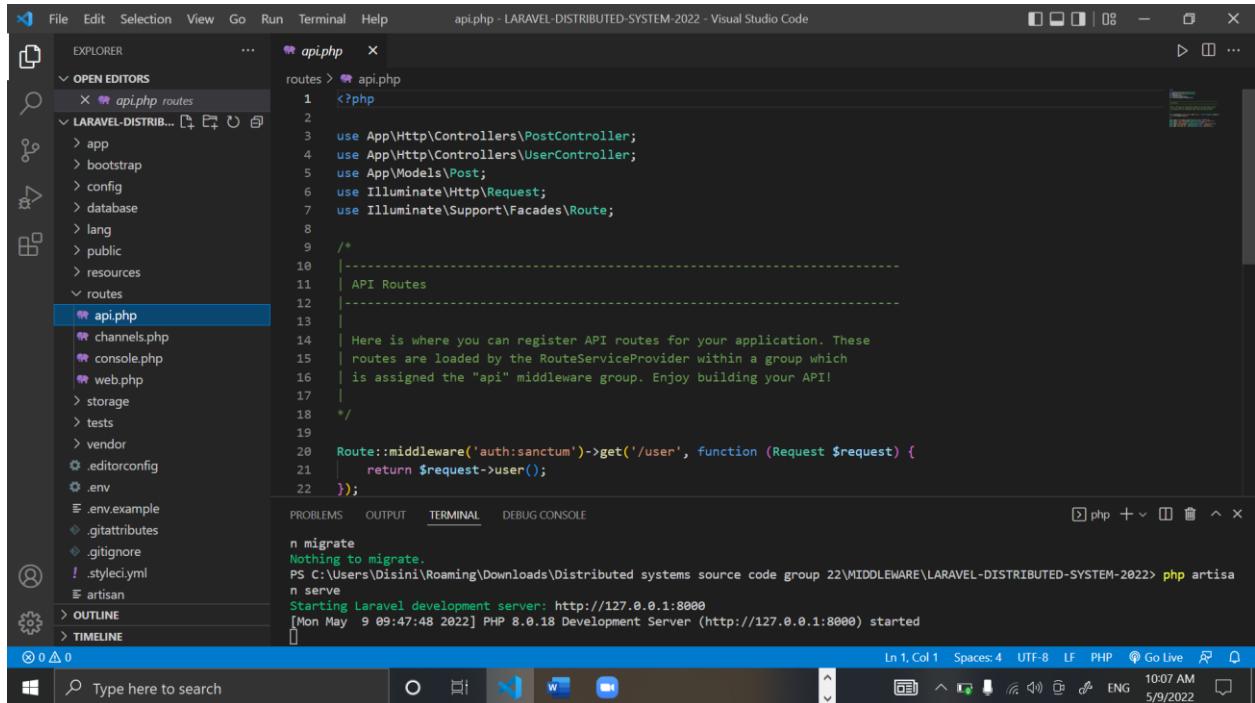
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\ MIDDLEWARE\ LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\ MIDDLEWARE\ LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started

Ln 1, Col 1 Spaces: 4 UTF-8 LF PHP ENG 10:06 AM 5/9/2022

Figure 32: UserController.php part 3

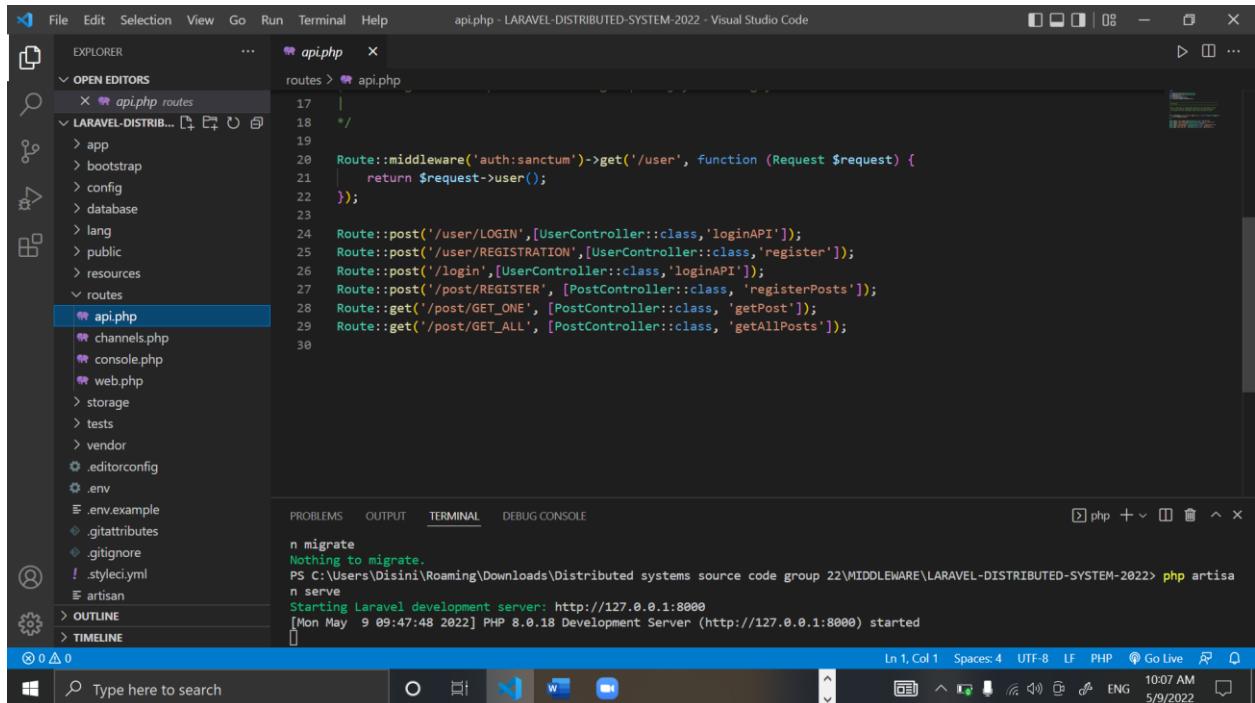
1.16.2 APIs



```
<?php
use App\Http\Controllers\PostController;
use App\Http\Controllers\UserController;
use App\Models\Post;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;

/*
|--------------------------------------------------------------------------
| API Routes
|--------------------------------------------------------------------------
|
| Here is where you can register API routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| is assigned the "api" middleware group. Enjoy building your API!
|
*/
Route::middleware('auth:sanctum')->get('/user', function (Request $request) {
    return $request->user();
});
```

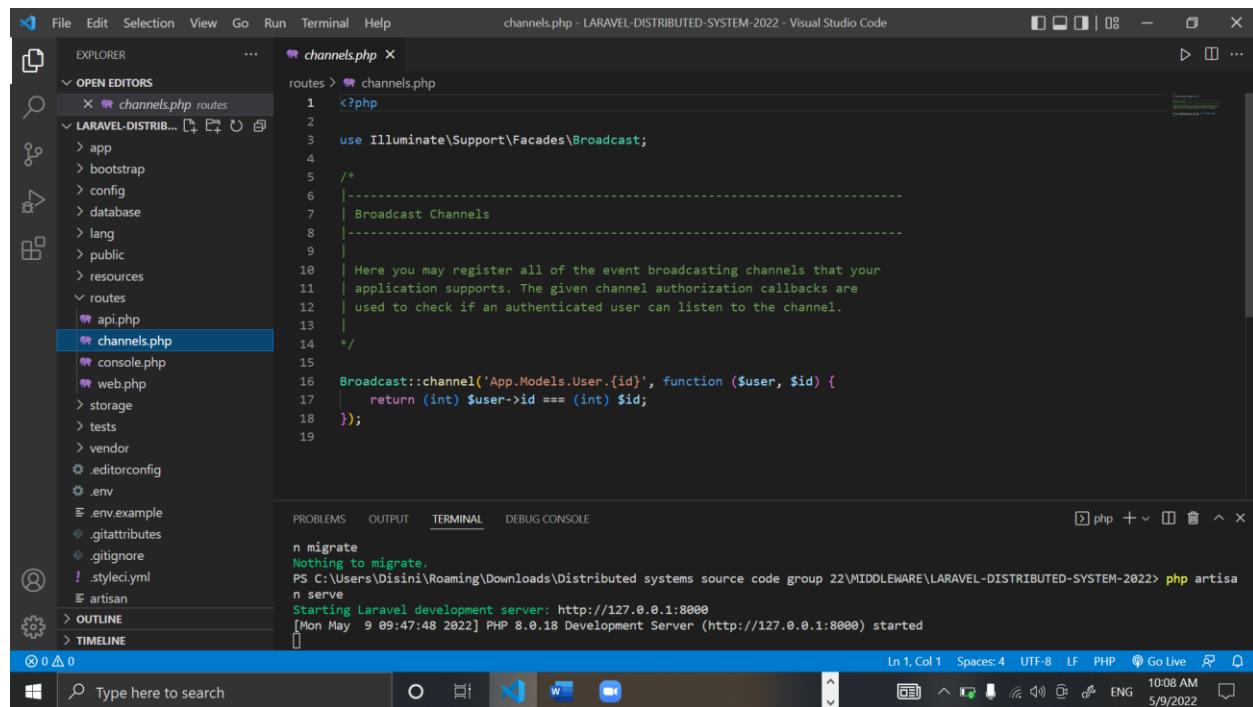
Figure 33: Api.php part 1



```
Route::middleware('auth:sanctum')->get('/user', function (Request $request) {
    return $request->user();
});

Route::post('/user/LOGIN',[UserController::class,'loginAPI']);
Route::post('/user/REGISTRATION',[UserController::class,'register']);
Route::post('/login',[UserController::class,'loginAPI']);
Route::post('/post/REGISTER', [PostController::class, 'registerPosts']);
Route::get('/post/GET_ONE', [PostController::class, 'getPost']);
Route::get('/post/GET_ALL', [PostController::class, 'getAllPosts']);
```

Figure 34: Api.php part 2



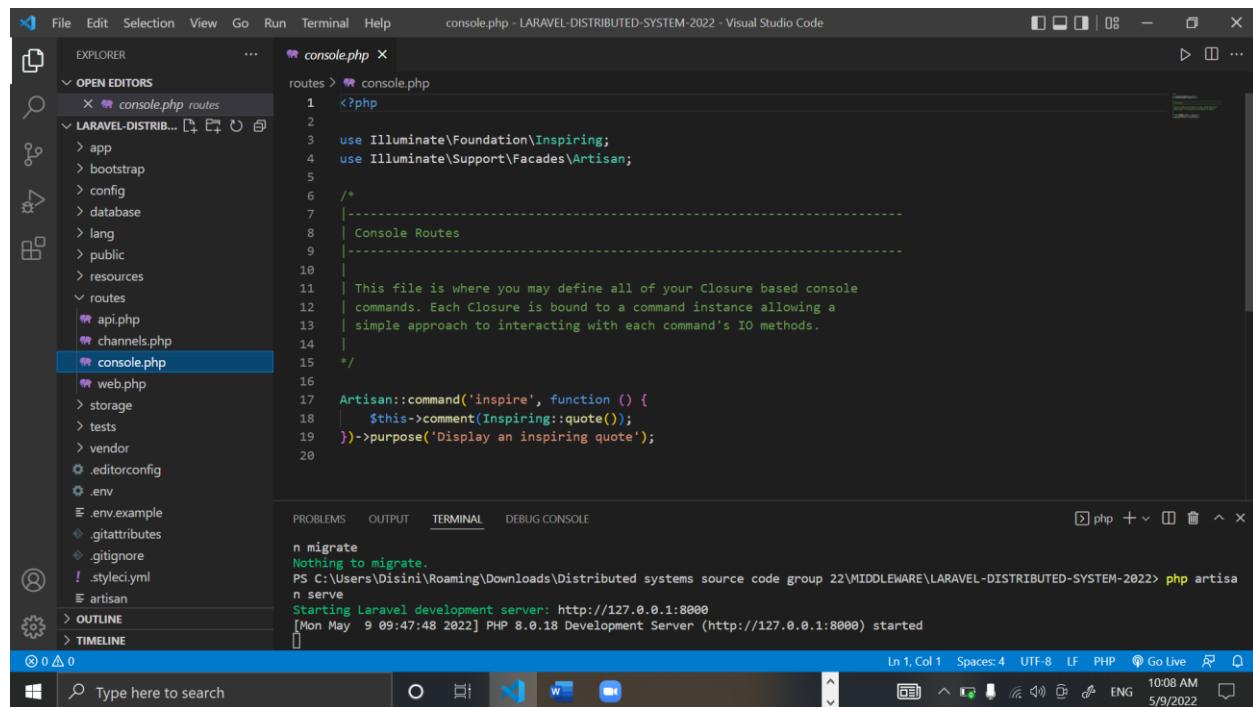
```
routes > channels.php
1 <?php
2
3 use Illuminate\Support\Facades\Broadcast;
4
5 /*
6 |-----
7 | Broadcast Channels
8 |-----
9 |
10 | Here you may register all of the event broadcasting channels that your
11 | application supports. The given channel authorization callbacks are
12 | used to check if an authenticated user can listen to the channel.
13 |
14 */
15
16 Broadcast::channel('App.Models.User.{id}', function ($user, $id) {
17     return (int) $user->id === (int) $id;
18 });
19
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
n migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan
n serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started
```

Ln 1, Col 1 Spaces:4 UTF-8 LF PHP ENG 10:08 AM 5/9/2022

Figure 35: Channels.php



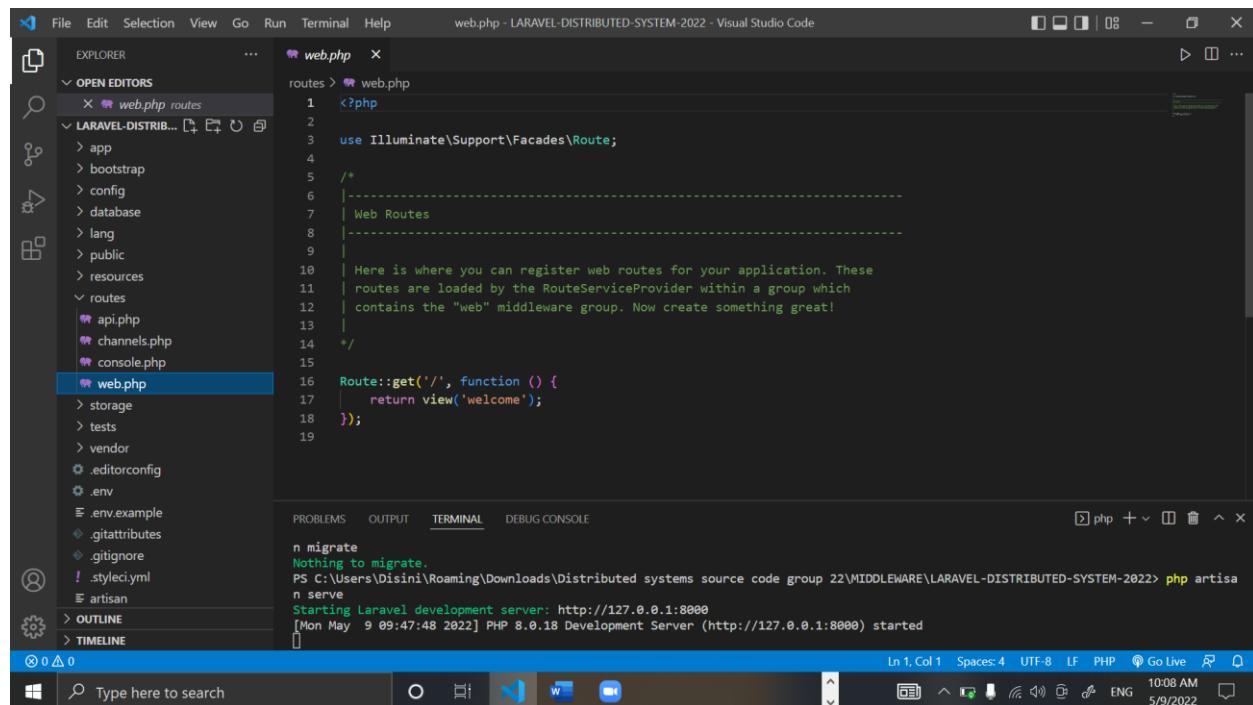
```
routes > console.php
1 <?php
2
3 use Illuminate\Foundation\Inspiring;
4 use Illuminate\Support\Facades\Artisan;
5
6 /*
7 |-----
8 | Console Routes
9 |-----
10 |
11 | This file is where you may define all of your Closure based console
12 | commands. Each Closure is bound to a command instance allowing a
13 | simple approach to interacting with each command's IO methods.
14 |
15 */
16
17 Artisan::command('inspire', function () {
18     $this->comment(Inspiring::quote());
19 })->purpose('Display an inspiring quote');
20
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
n migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan
n serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started
```

Ln 1, Col 1 Spaces:4 UTF-8 LF PHP ENG 10:08 AM 5/9/2022

Figure 36: Console.php



File Edit Selection View Go Run Terminal Help web.php - LARAVEL-DISTRIBUTED-SYSTEM-2022 - Visual Studio Code

routes > web.php

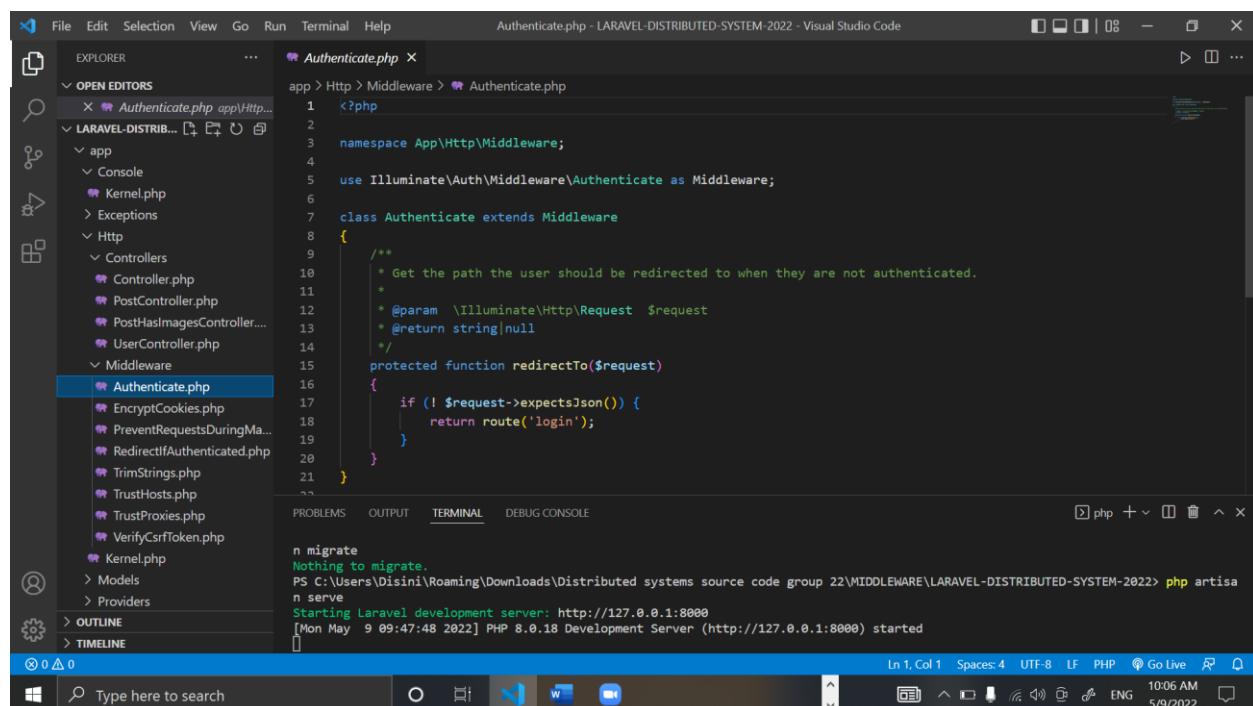
```
1 <?php
2
3 use Illuminate\Support\Facades\Route;
4
5 /*
6 |-----|
7 | Web Routes
8 |-----|
9 |
10 | Here is where you can register web routes for your application. These
11 | routes are loaded by the RouteServiceProvider within a group which
12 | contains the "web" middleware group. Now create something great!
13 |
14 */
15
16 Route::get('/', function () {
17     return view('welcome');
18 });
19
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
n migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan
n serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started
```

Ln 1, Col 1 Spaces:4 UTF-8 LF PHP ENG 10:08 AM 5/9/2022

Figure 37: Web.php



File Edit Selection View Go Run Terminal Help Authenticate.php - LARAVEL-DISTRIBUTED-SYSTEM-2022 - Visual Studio Code

app > Http > Middleware > Authenticate.php

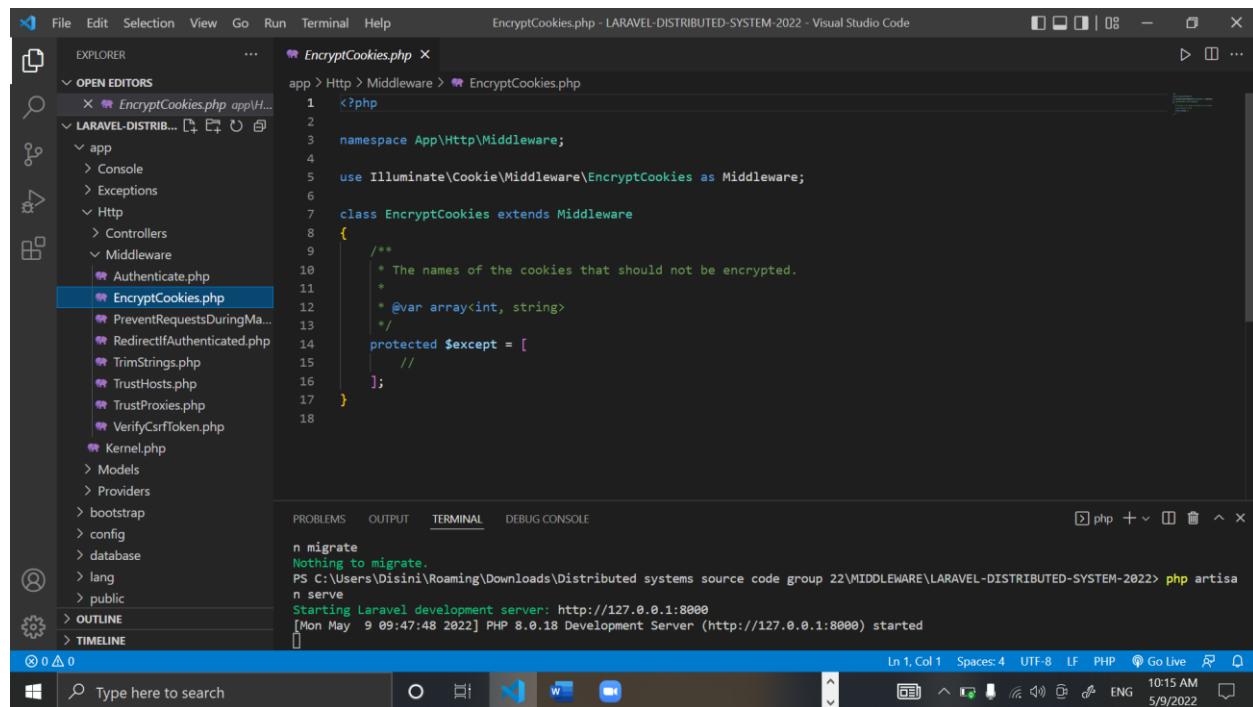
```
1 <?php
2
3 namespace App\Http\Middleware;
4
5 use Illuminate\Auth\Middleware\Authenticate as Middleware;
6
7 class Authenticate extends Middleware
8 {
9
10     /**
11      * Get the path the user should be redirected to when they are not authenticated.
12      *
13      * @param \Illuminate\Http\Request $request
14      * @return string|null
15      */
16     protected function redirectTo($request)
17     {
18         if (! $request->expectsJson()) {
19             return route('login');
20         }
21     }
22 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
n migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan
n serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started
```

Ln 1, Col 1 Spaces:4 UTF-8 LF PHP ENG 10:06 AM 5/9/2022

Figure 38: Authenticate.php

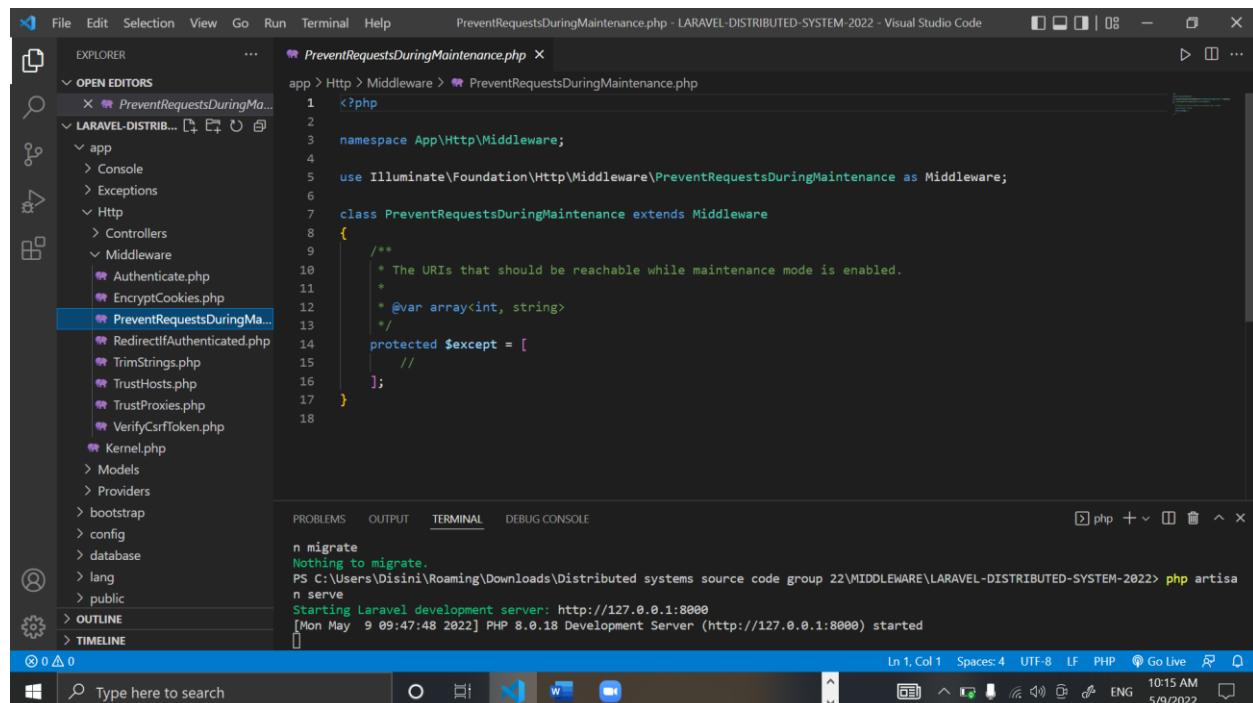


```
EncryptCookies.php
1 <?php
2
3 namespace App\Http\Middleware;
4
5 use Illuminate\Cookie\Middleware\EncryptCookies as Middleware;
6
7 class EncryptCookies extends Middleware
8 {
9
10     /**
11      * The names of the cookies that should not be encrypted.
12      *
13      * @var array<int, string>
14      */
15     protected $except = [
16
17     ];
18 }
```

TERMINAL

```
n migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan
n serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started
```

Figure 39: EncryptCookies.php

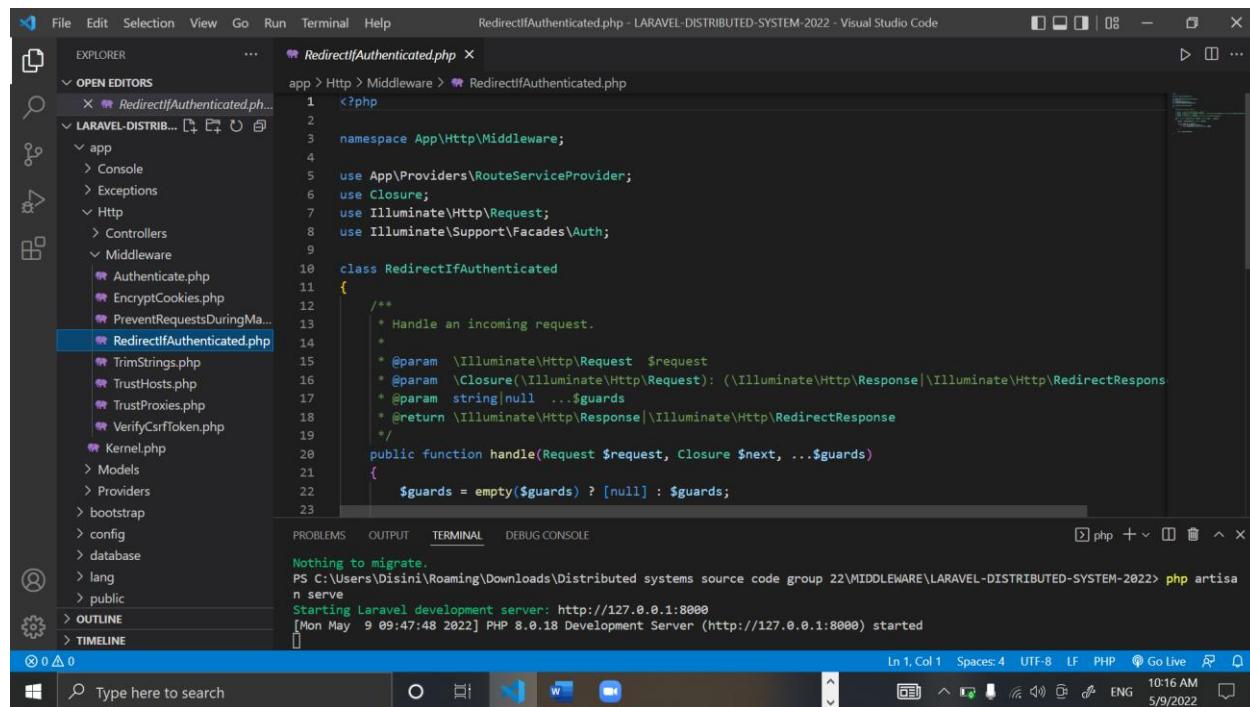


```
PreventRequestsDuringMaintenance.php
1 <?php
2
3 namespace App\Http\Middleware;
4
5 use Illuminate\Foundation\Http\Middleware\PreventRequestsDuringMaintenance as Middleware;
6
7 class PreventRequestsDuringMaintenance extends Middleware
8 {
9
10     /**
11      * The URIs that should be reachable while maintenance mode is enabled.
12      *
13      * @var array<int, string>
14      */
15     protected $except = [
16
17     ];
18 }
```

TERMINAL

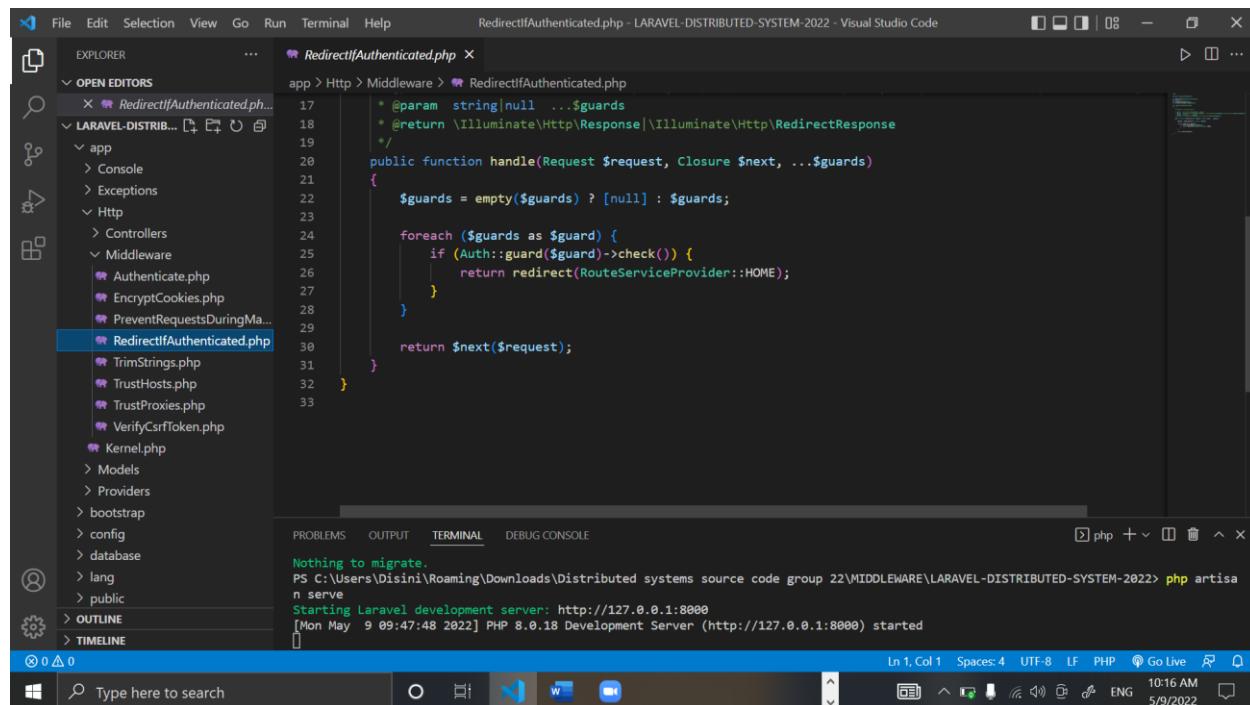
```
n migrate
Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan
n serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started
```

Figure 40: Preventrequestsduringmaintenance.php



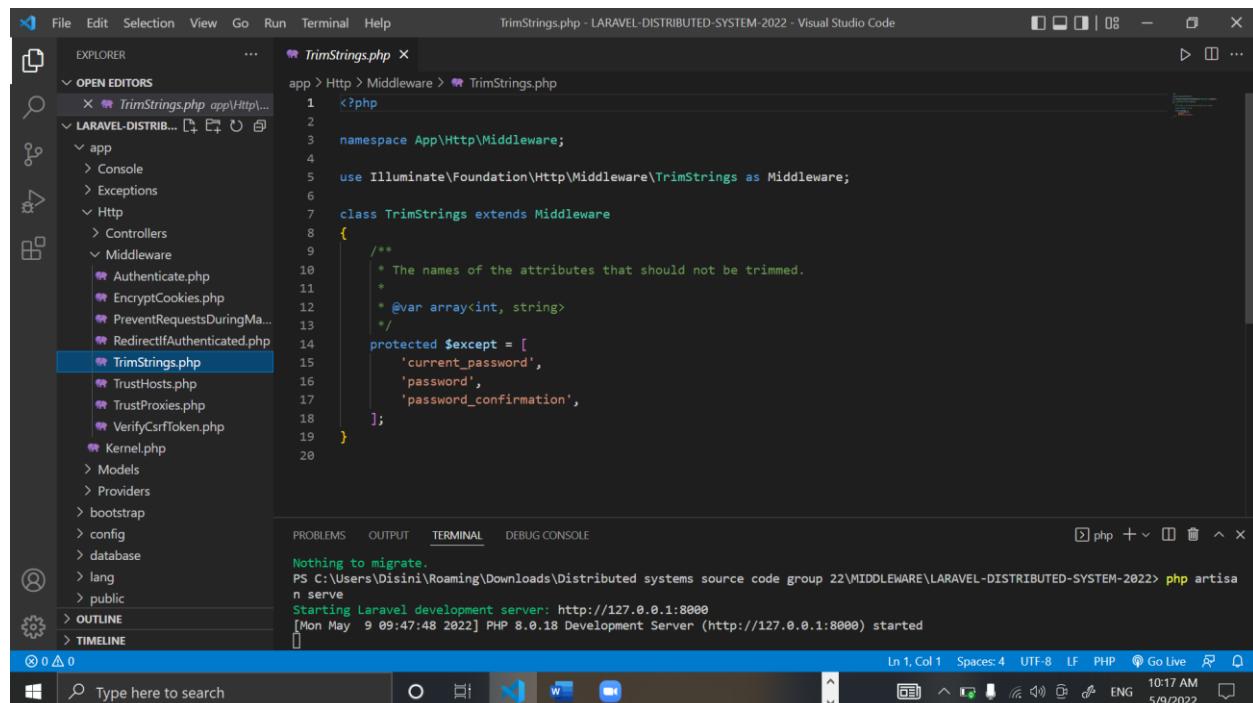
```
1 <?php
2
3 namespace App\Http\Middleware;
4
5 use App\Providers\RouteServiceProvider;
6 use Closure;
7 use Illuminate\Http\Request;
8 use Illuminate\Support\Facades\Auth;
9
10 class RedirectIfAuthenticated
11 {
12     /**
13      * Handle an incoming request.
14      *
15      * @param \Illuminate\Http\Request $request
16      * @param \Closure(\Illuminate\Http\Request): (\Illuminate\Http\Response|\Illuminate\Http\RedirectResponse) $next
17      * @param string|null ...$guards
18      * @return \Illuminate\Http\Response|\Illuminate\Http\RedirectResponse
19      */
20     public function handle(Request $request, Closure $next, ...$guards)
21     {
22         $guards = empty($guards) ? [null] : $guards;
23     }
24
25     /**
26      * @param string|null ...$guards
27      * @return \Illuminate\Http\Response|\Illuminate\Http\RedirectResponse
28      */
29     public function handle(Request $request, Closure $next, ...$guards)
30     {
31         $guards = empty($guards) ? [null] : $guards;
32
33         foreach ($guards as $guard) {
34             if (Auth::guard($guard)->check()) {
35                 return redirect(RouteServiceProvider::HOME);
36             }
37         }
38
39         return $next($request);
40     }
41 }
```

Figure 41: RedirectIfAuthenticated.php part 1



```
17     /**
18      * @param string|null ...$guards
19      * @return \Illuminate\Http\Response|\Illuminate\Http\RedirectResponse
20      */
21     public function handle(Request $request, Closure $next, ...$guards)
22     {
23         $guards = empty($guards) ? [null] : $guards;
24
25         foreach ($guards as $guard) {
26             if (Auth::guard($guard)->check()) {
27                 return redirect(RouteServiceProvider::HOME);
28             }
29         }
30
31         return $next($request);
32     }
33 }
```

Figure 42: RedirectIfAuthenticated.php part 2

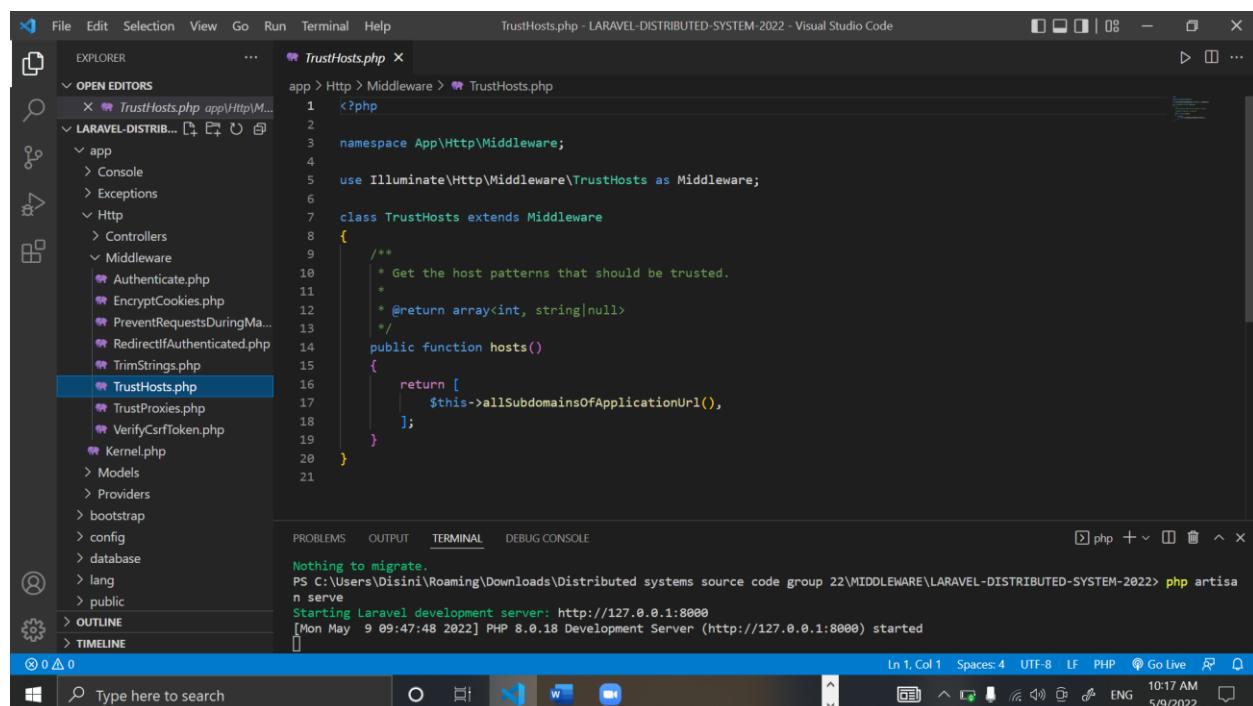


```
TrimStrings.php
1 <?php
2
3 namespace App\Http\Middleware;
4
5 use Illuminate\Foundation\Http\Middleware\TrimStrings as Middleware;
6
7 class TrimStrings extends Middleware
8 {
9
10     /**
11      * The names of the attributes that should not be trimmed.
12      *
13      * @var array<int, string>
14      */
15
16     protected $except = [
17         'current_password',
18         'password',
19         'password_confirmation',
20     ];
21 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started

Figure 43: TrimStrings.php

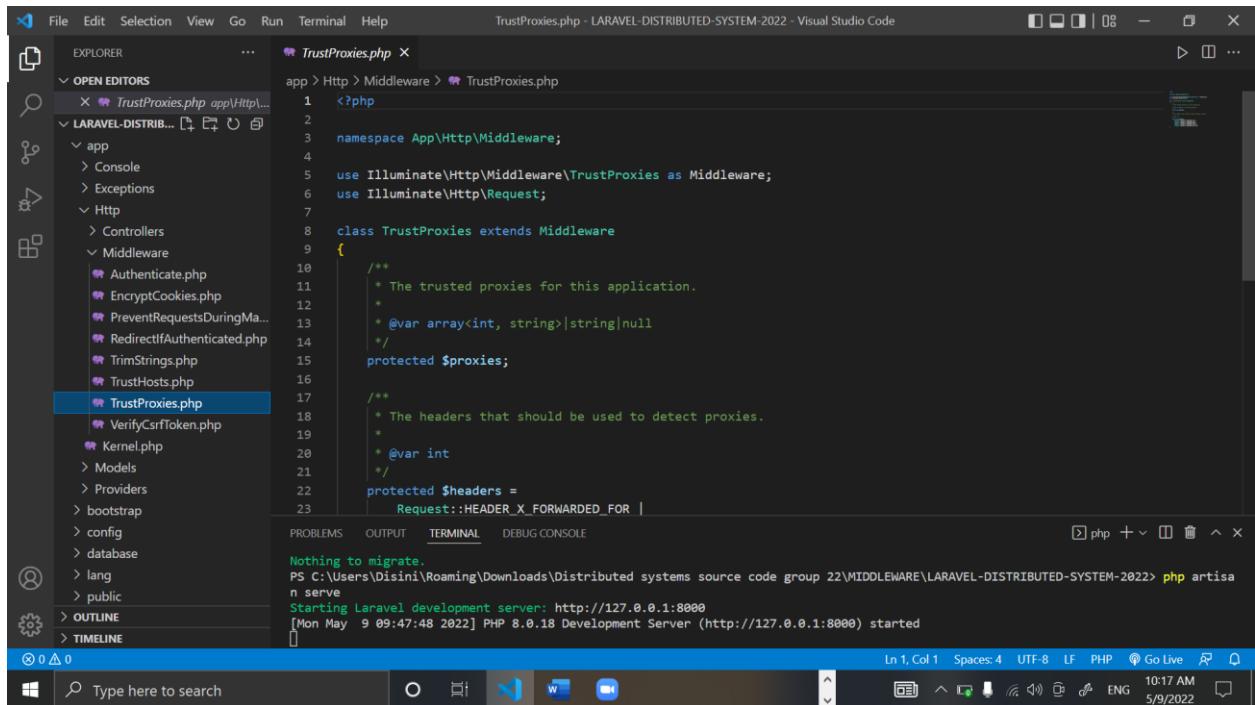


```
TrustHosts.php
1 <?php
2
3 namespace App\Http\Middleware;
4
5 use Illuminate\Http\Middleware\TrustHosts as Middleware;
6
7 class TrustHosts extends Middleware
8 {
9
10     /**
11      * Get the host patterns that should be trusted.
12      *
13      * @return array<int, string|null>
14      */
15
16     public function hosts()
17     {
18         return [
19             $this->allSubdomainsOfApplicationUrl(),
20         ];
21     }
22 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\MIDDLEWARE\LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started

Figure 44: TrustsHosts.php



```
1 <?php
2
3 namespace App\Http\Middleware;
4
5 use Illuminate\Http\Middleware\TrustProxies as Middleware;
6 use Illuminate\Http\Request;
7
8 class TrustProxies extends Middleware
9 {
10     /**
11      * The trusted proxies for this application.
12      *
13      * @var array<int, string>|string|null
14      */
15     protected $proxies;
16
17     /**
18      * The headers that should be used to detect proxies.
19      *
20      * @var int
21      */
22     protected $headers =
23         Request::HEADER_X_FORWARDED_FOR |
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

Nothing to migrate.

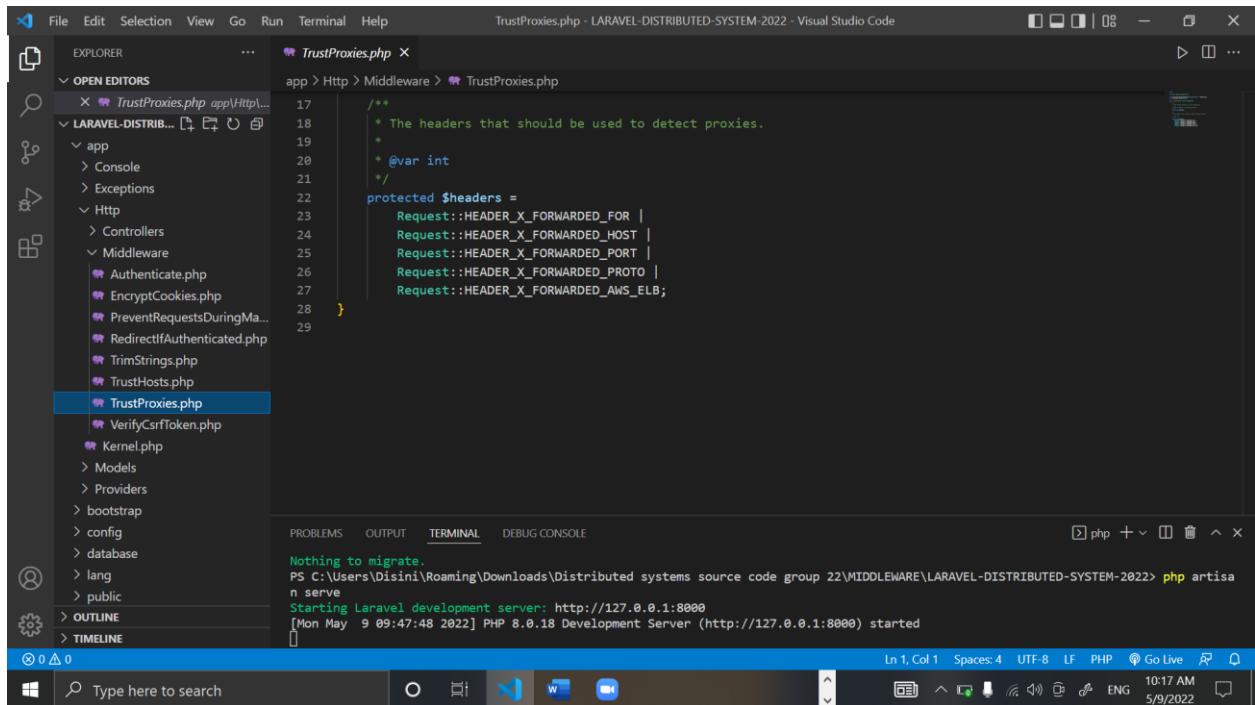
PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\ MIDDLEWARE\ LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan serve

Starting Laravel development server: http://127.0.0.1:8000

[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started

Ln 1, Col 1 Spaces:4 UTF-8 LF PHP ENG 10:17 AM 5/9/2022

Figure 45: Trustproxies.php part 1



```
17 /**
18  * The headers that should be used to detect proxies.
19  *
20  * @var int
21  */
22 protected $headers =
23     Request::HEADER_X_FORWARDED_FOR |
24     Request::HEADER_X_FORWARDED_HOST |
25     Request::HEADER_X_FORWARDED_PORT |
26     Request::HEADER_X_FORWARDED_PROTO |
27     Request::HEADER_X_FORWARDED_AWS_ELB;
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

Nothing to migrate.

PS C:\Users\Disini\Roaming\Downloads\ Distributed systems source code group 22\ MIDDLEWARE\ LARAVEL-DISTRIBUTED-SYSTEM-2022> php artisan serve

Starting Laravel development server: http://127.0.0.1:8000

[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started

Ln 1, Col 1 Spaces:4 UTF-8 LF PHP ENG 10:17 AM 5/9/2022

Figure 46: trustproxies.php part 2

```

1 <?php
2
3 namespace App\Http\Middleware;
4
5 use Illuminate\Foundation\Http\Middleware\VerifyCsrfToken as Middleware;
6
7 class VerifyCsrfToken extends Middleware
8 {
9
10     /**
11      * The URIs that should be excluded from CSRF verification.
12      *
13      * @var array<int, string>
14      */
15     protected $except = [
16         //
17     ];
18 }

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

Nothing to migrate.
PS C:\Users\Disini\Roaming\Downloads\distributed systems source code group 22\middleware\laravel-distributed-system-2022> php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Mon May 9 09:47:48 2022] PHP 8.0.18 Development Server (http://127.0.0.1:8000) started

Figure 47: VerifyCsrfToken.php

Future enhancement of the system

We as a group would love to upgrade our project if we are to continue this project, this project is to be used by any person who has access to the internet. With this system people can focus and entail in their safety.

We would like to add features were the user can live forecasts and, we would love to add reports from weather stations into the systems, were people can be updated. We would also like to enhance the project more with checking for weather patterns and collecting data for the betterment of human life.

Risk assessment

Risk	Management Strategy
Risk of Data Loss/ Potential Application Failure	Backup and copy on a storage device.
Overrun on the schedule	The project plan now includes a contingency plan. Highlights and supervisor meetings will be used to track and monitor progress. More days have been added to the estimated completion dates for each step.
Complexity of the project	To adhere to a set of rules that would allow the basic functionality to be implemented and a working product to be produced.
Problems with the technology needed to complete the project	A basic prototype will be constructed for each phase to examine and validate validity. Maintaining a record of each stage.
Other requirements	There will be other projects operating concurrently, and this will receive full attention. A time management strategy would assist me in staying on track.
Lack of information on this topic	<ul style="list-style-type: none"> Before starting, do a feasibility study on the topic. Read books, essays, etc. on the topic.

	<ul style="list-style-type: none"> • Research on the internet.
Bad Timing	<ul style="list-style-type: none"> • Plan, then stick to it. • You must determine which project stages or components are most important to you, as well as the short, medium, and long-term impact of each stage/component.
Poor Code Quality	<ul style="list-style-type: none"> • Review of the code • Coding rules and guidelines that are easy to understand • All code is evaluated. • The Method of Work.

Quality plan

Unit Testing	Separate platforms and code blocks were evaluated in debug mode, bug fixing, and problems were resolved
Integrated Testing	In unit testing as shown with the screen shots of the working application the issues faced with the collaborative workload of the application were fixed and resolved

Table 4: Quality Plan

Summary

To sum everything up, the team has successfully fulfilled the necessary requirements. In comparison to the project proposal, there are some minor changes and deviations in consonance to the utilized programming languages, IDE's and Frameworks. Yet, it is evident that the distributed system has successfully covered the mandatory concepts of building a distributed system.

We have used Laravel to implement our load balancer and middleware.

The overall distributed system is responsive and easy to use. When it comes to fault tolerance, potential network failure is reduced by the addition of one server.

Within the report, we have clearly specified the individual contributions, Architectural Diagrams, middleware and API.

Individual contribution

Member	Index Number	Contribution
J.A. Mujeeb	10707284	I created the dashboard and login in the standalone application created in NetBeans. I also assisted with creating the routes for the API.
P.L. Dilhani	10709402	I created the Load Posts part in the NetBeans application. I also assisted in creating the middleware in the API.
M.D.A. Medhavi	10707278	I created the showing the weather in the web application and the login was done by me. I also assisted in creating the config files.
S.O. Perera	10707315	I created the part where the user can see the posts in the web application and assisted in the creation of config files.
N.S De Alwis	10707160	I created the blog post page in mobile application and dashboard section where users are shown the list of posts related to whether, including the development of the Adapters and Models folders I also assisted in the creation of controllers.

G.M.D.D. Ratnayake	10707351	I created the Signup, Login, and the splash screen in the mobile application, including the development of the Adapters and Models folders. I also assisted in creating the models in the API.
-----------------------	----------	--

References

- <https://forge.laravel.com/docs/1.0/servers/load-balancing.html>
https://www.w3schools.com/java/java_getstarted.asp
<https://laravel.com/docs/9.x/middleware>
<https://kinsta.com/blog/install-php/>
<https://medium.com/weblineindia/8-reasons-to-use-java-for-mobile-app-development-87bed4544203#:~:text=The%20primary%20benefit%20of%20using,is%20extensible%2C%20scalable%20and%20adaptable.>
Slideshare.net. 2022. Weather Now. [Online] Available at: <<https://www.slideshare.net/AmilaWijayarathna/weather-now-75943627>> [Accessed 8 May 2022].
<https://cms.artio.net/en/what-we-do/software-analysis>
https://www.tutorialspoint.com/software_engineering/software_design_basics.htm
https://www.youtube.com/results?search_query=how+to+create+a+distributed+system
<https://www.javatpoint.com/software-project-planning>
<https://www.youtube.com/watch?v=U9P96XQ2688>