



# Final Project Report

Subject: Web Technologies - II UE17CS353

Project Title: APP Gaming

Project Team: Pradyumna YM

Pruthvish E

Anush V Kini

PES1201700986

PES1201701629

PES1201701646





# Project Description

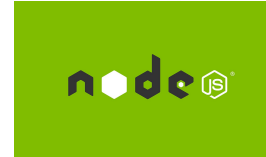
- We have implemented a one stop gaming website using Angular with the following features.
  - Play Games
  - Login System with authentication using JWT and bcryptJS
  - Friends System
  - Global Chat
  - Multiplayer Real Time Gaming (for Chess)
  - RSS feeds
  - Synergy scores (recommender system)
  - Smart profile ordering (using transfer learning CNNs)
  - REST API in backend using Nodejs
  - All of them developed from scratch by us.





# Technologies Used

- Programming Languages
  - Python
  - Typescript/Javascript
- Tools
  - Angular
  - Node JS
  - Tensorflow 2.0
  - Bootstrap
  - Mongodb



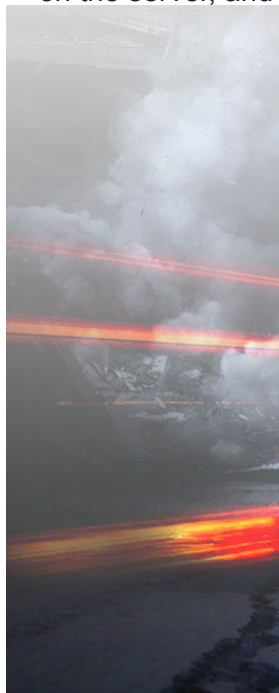
Bootstrap 4





## Techniques Implemented

- **RSS feeds** for the blog section of our website. We obtain the RSS feed from a gaming website, parse it on the server, and serve JSON objects to the client to display the latest information.



**Final Fantasy 7 Remake Chapter 11 Walkthrough: Haunted (Spoiler-Free)**

Wed, 15 Apr 2020 23:08:00 -0700

We're here to continue guiding you through Midgar in [Final Fantasy 7 Remake](#) with another installment of our spoiler-free walkthrough. We're venturing into the Sector 7 Train Graveyard in Chapter 11, a section of the game that includes some especially useful weapons to find, but which are some of the easiest in the game to miss. Keep reading below for all the details.


Check out the rest of our [Final Fantasy 7 Remake guides](#) so you don't miss a single secret or collectible. You can also [read our FF7 Remake review](#).

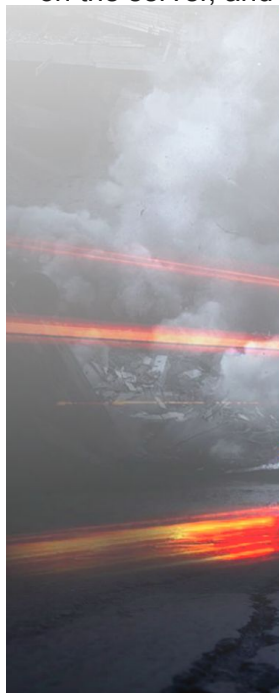
- Chapter 1: The Destruction of Reactor 1
- Chapter 2: Fateful Encounters
- Chapter 3: Home Sweet Slum
- Chapter 4: Mad Dash
- Chapter 5: Dogged Pursuit
- Chapter 6: Light The Way
- Chapter 7: A Trap Is Sprung
- Chapter 8: Budding Bodyguard
- Chapter 9: The Town That Never Sleeps
- Chapter 10: Rough Waters

**Chapter 11 Walkthrough: Haunted**

Enter the train graveyard and pass through the first train car. You'll exit out into a large area with a few enemies to kill. Go straight across to the east and pass through another train car, then head south behind a cargo container to **find a chest with a Mythril Rod weapon** for Aerith.

[Continue Reading at GameSpot](#)





**League Of Legends Is Balancing Jungling In Patch 10.8**


Wed, 15 Apr 2020 23:07:00 -0700

The latest update for [League of Legends](#) seeks to bring some new changes to the jungle position, evening out the weakness of the junglers in early game and nerfing the late game power these positions tend to bring. For a full rundown on 10.8, check out the [developer's blog](#).

Riot is seeking to restructure the jungle position, changing up the aggressive tactics that some junglers use during the start of the matches, instead offering bigger rewards for farming in early game. Extra sustain is also being added to allow those who fall to low health in the jungle, keeping a jungler alive in a tough position.

According to the developers, "Junglers are appropriately influential as a role, but we want to reduce the perception that they have to have an aggressive early presence to be effective. Instead, we're giving them opportunities for more personal power through proper farming as the game progresses."

[Continue Reading at GameSpot](#)



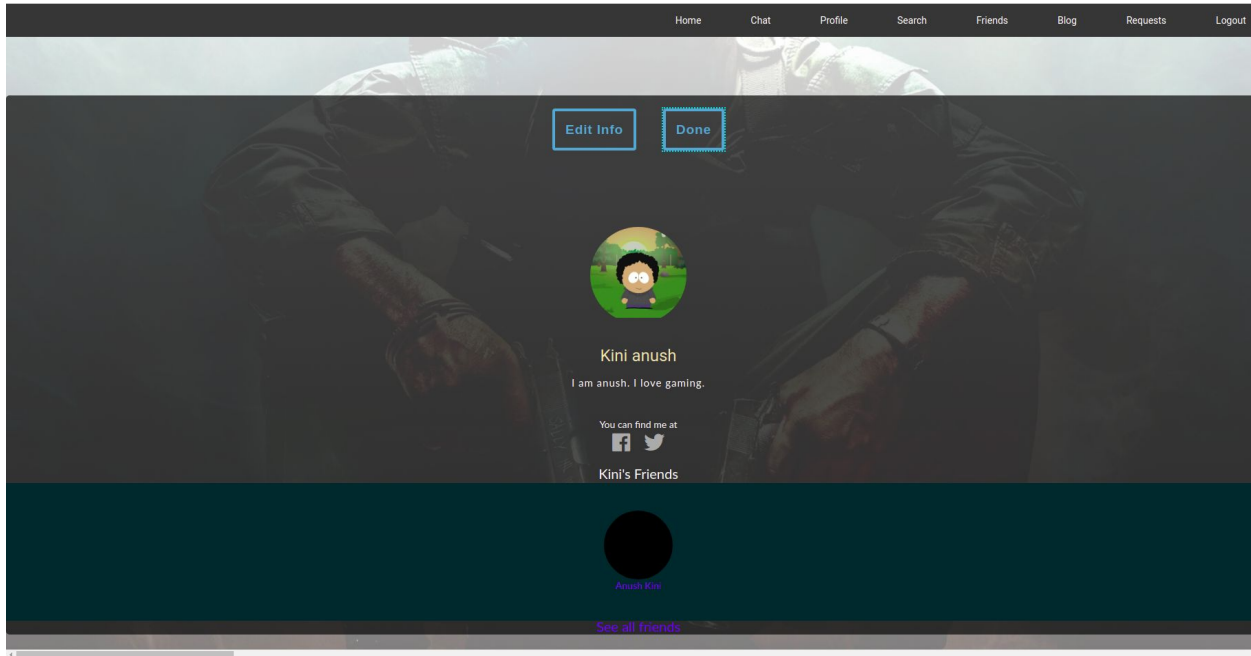






## Techniques Implemented

- **Multistage Download** : profile pictures route of the client uses multistage download, where the details of the user is retrieved from the database first, and then the profile picture of the user is retrieved, and in the next stage, his friend profile details are retrieved, and lastly, his friend's profile pictures are obtained.

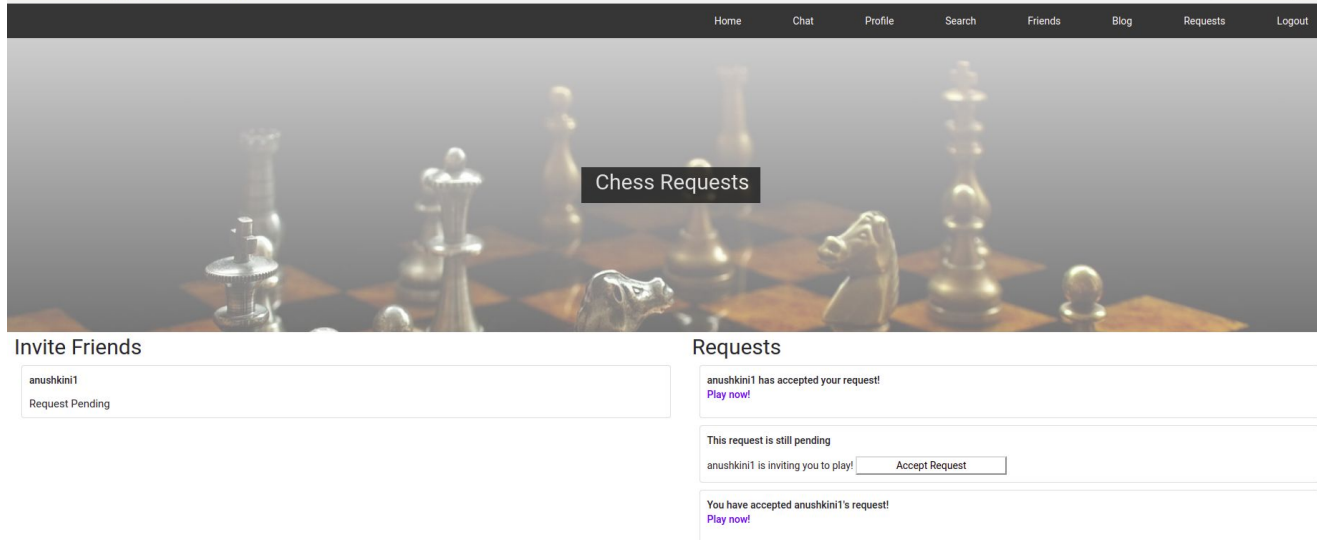






## Techniques Implemented

- **Periodic refresh** : used in the chess requests component, where the status of the game requests get updated every 5 seconds.



The screenshot shows a web application interface for chess requests. The top navigation bar includes links for Home, Chat, Profile, Search, Friends, Blog, Requests, and Logout. The main header features a chessboard background with the text "Chess Requests". Below this, there are two main sections: "Invite Friends" and "Requests".

**Invite Friends**

anushkini1  
Request Pending

**Requests**

anushkini1 has accepted your request!  
[Play now!](#)

This request is still pending  
anushkini1 is inviting you to play!

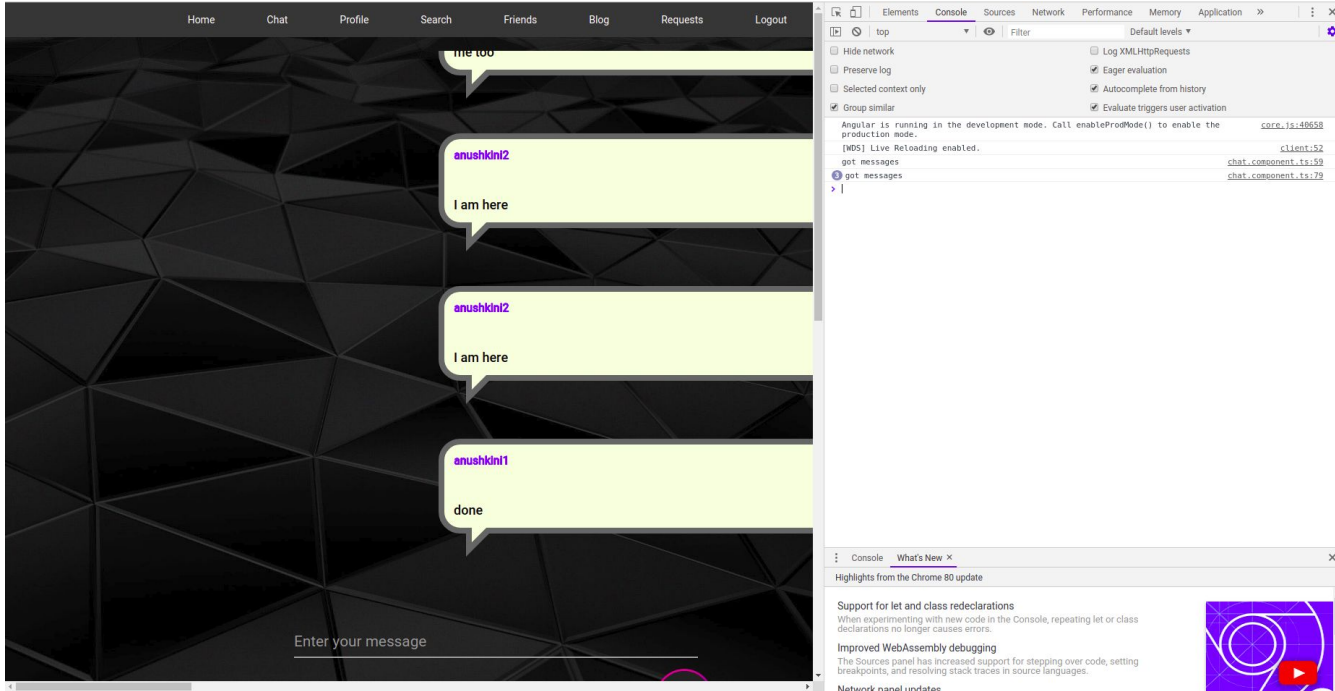
You have accepted anushkini1's request!  
[Play now!](#)





## Techniques Implemented

- **Periodic refresh** : also used in the chat section of the website.







## Techniques Implemented

- **REST API:** our nodejs backend application makes use of a stateless REST API that serves the required data to the client in JSON format.

```
userRouter.js
71
72 router.get("/userdetails", verifyToken, (req, res) => {
73   User.getUserByUserName(req.user.username, (err, user) =>{
74     if(err) return res.sendStatus(500);
75     if(!user) return res.sendStatus(404);
76     var {img, firstname, lastname, username, email, facebook, twitter, summary} = user;
77     return res.status(200).json({firstname, lastname, username, img, email, facebook, twitter, summary});
78   });
79 });
80
81 router.get("/profile/username", verifyToken, (req, res) =>{
82   User.getUserByUserName(req.params.username, (err, user) =>{
83     if(err) return res.sendStatus(500);
84     if(!user) return res.sendStatus(404);
85     var {img, firstname, lastname, username, email, facebook, twitter, summary} = user;
86     return res.status(200).json({img, firstname, lastname, username, email, facebook, twitter, summary});
87   });
88 });
89
90 router.post("/friendslist", verifyToken, (req, res) =>{
91   console.log(req.body);
92   User.getUserByUserName(req.body.username, (err, user) =>{
93     if(err) return res.sendStatus(500);
94     if(!user) return res.sendStatus(404);
95     var {friends} = user;
96     return res.status(200).json(friends);
97   });
98 });
99
100 router.get("/friendrequests", verifyToken, (req, res) =>{
101   User.getUserByUserName(req.user.username, (err, user) =>{
102     if(err) return res.sendStatus(500);
103     if(!user) return res.sendStatus(404);
104     var {friendrequests} = user;
105     return res.status(200).json(friendrequests);
106   });
107 });
108
109 router.post("/sendrequest", verifyToken, (req, res) =>{
110   if(!req.body.username) return res.sendStatus(405);
111   User.getUserByUserName(req.body.username, (err, user) =>{
112     if(err) return res.sendStatus(500);
113     if(!user) return res.sendStatus(404);
114   });
115 });
116
117 //Desktop\hw2\project\routes\userRouter.js 1:1
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
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```





## Intelligent Functionality

**Smart friend reordering for profiles:** We have implemented a convolutional neural network to extract features from users' profile pictures. Based on the euclidean distance of the features of the friend profile pictures, the friends list is ordered.



Initial Ordering





## Intelligent Functionality

**Smart friend reordering for profiles:** We have implemented a convolutional neural network to extract features from users' profile pictures. Based on the euclidean distance of the features of the friend profile pictures, the friends list is ordered.



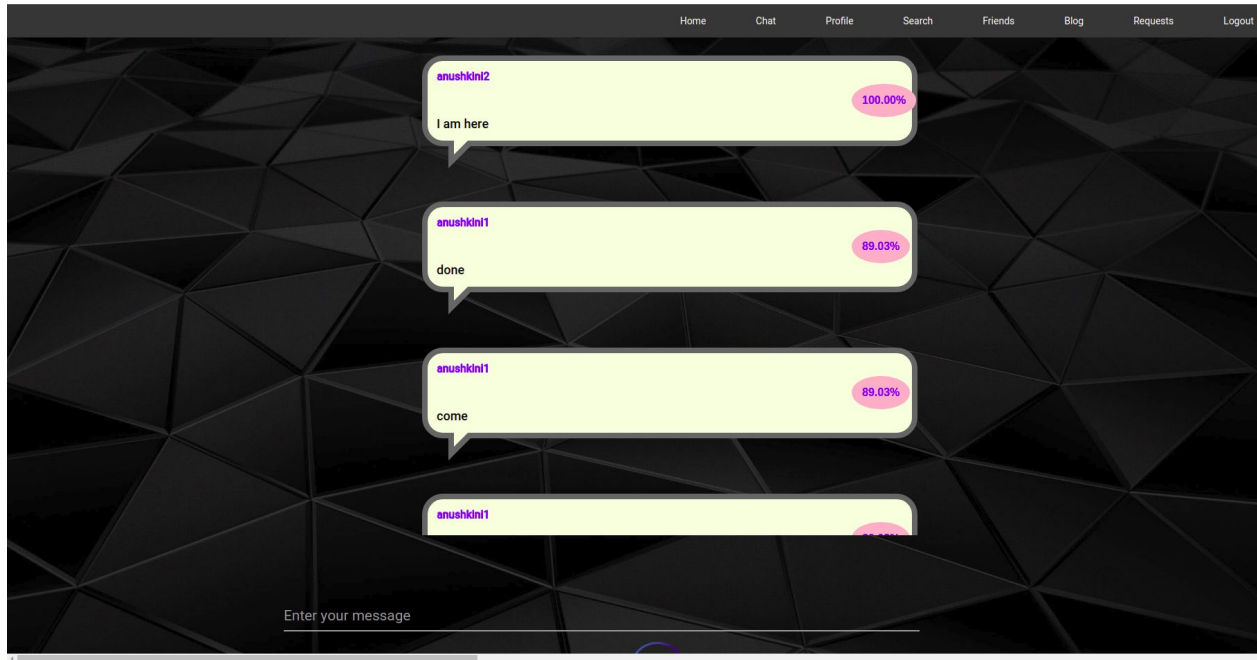
After running script





## Intelligent Functionality

**Synergy Indicator:** We have developed a smart synergy indicator for the chat section of our website, which gives users a percentage of their synergy with another user. This can be used to make friends who play the same games as you and as a fun indicator of how similar you are to another user.







Q / A

