



Duo: AI for GitLab

LC User Meeting
March 13th, 2025

John Consolati

Prepared by LLNL under Contract DE-AC52-07NA27344.



What is it?

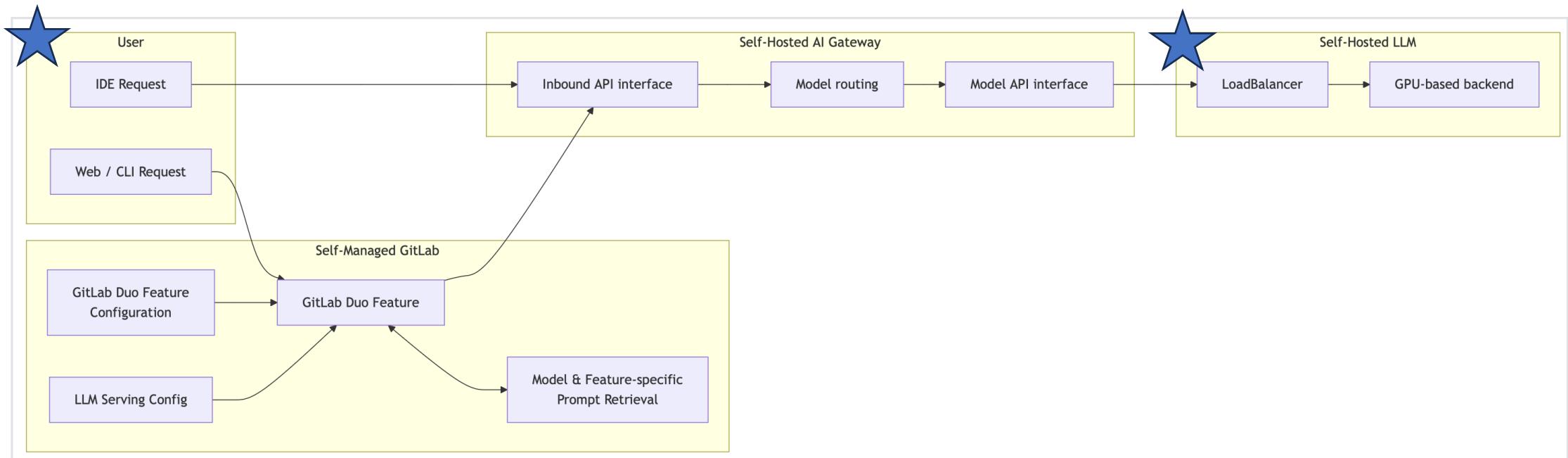
- New GitLab functionality that lets GitLab talk to an LLNL-managed large language model
- AI inference capabilities are accessible across the GitLab web UI, GitLab web IDE, and various desktop IDE plugins
- GitLab context-aware AI enables [many features](#):
 - Today: chat, code suggestions, code explanation, code refactoring, code fixes, test code generation
 - Future: Summarize discussions in issues/MRs, generate MRs automatically, root cause analysis for CI pipeline failures, code vulnerability analysis



Where is it?

- Deployed as Early Access to CZ GitLab late January 2025, announced via [Tech Bulletin #576](#)
- LC purchased 100 seats for the pilot, and there are seats remaining for interested users. Contact lc-gitlab-admin@llnl.gov
- Exploring possibilities for RZ deployment (soon) and SCF (later). Details in subsequent slides.

Component Architecture



Source: gitlab.com

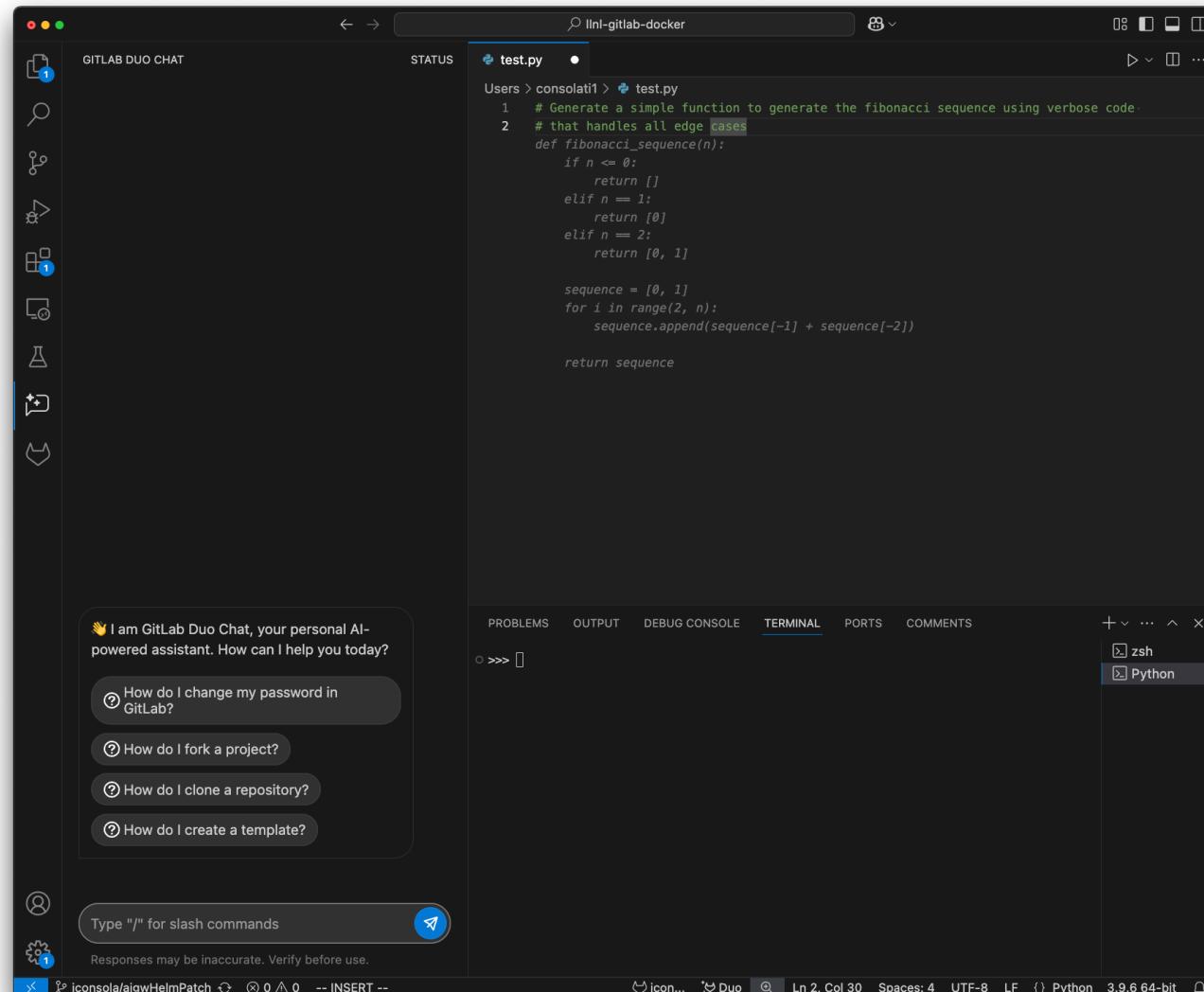
- Duo introduces several new components, but this quick primer will focus on the end-user experience (starred). Where can I use it, and what models are backing it?



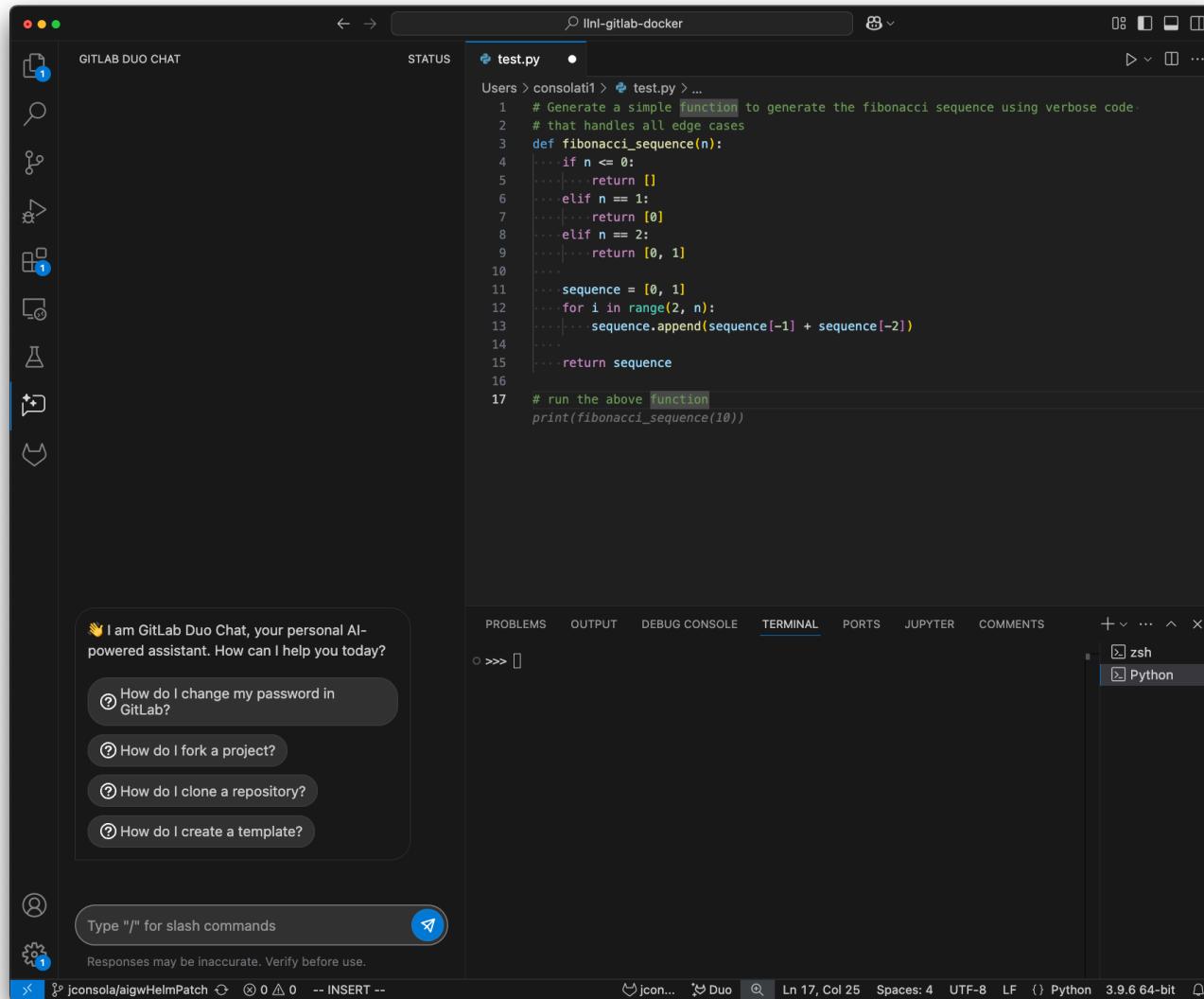
User Desktop IDE Integration

- GitLab [provides plugins](#) for various desktop IDEs such as VSCode, JetBrains, Visual Studio
- Plugins authenticate to CZ GitLab using a Personal Access Token (PAT) over 443/HTTPS
 - Same PAT as today; subject to 30-day expiration

Desktop IDE – Example Workflow



Desktop IDE – Example Workflow



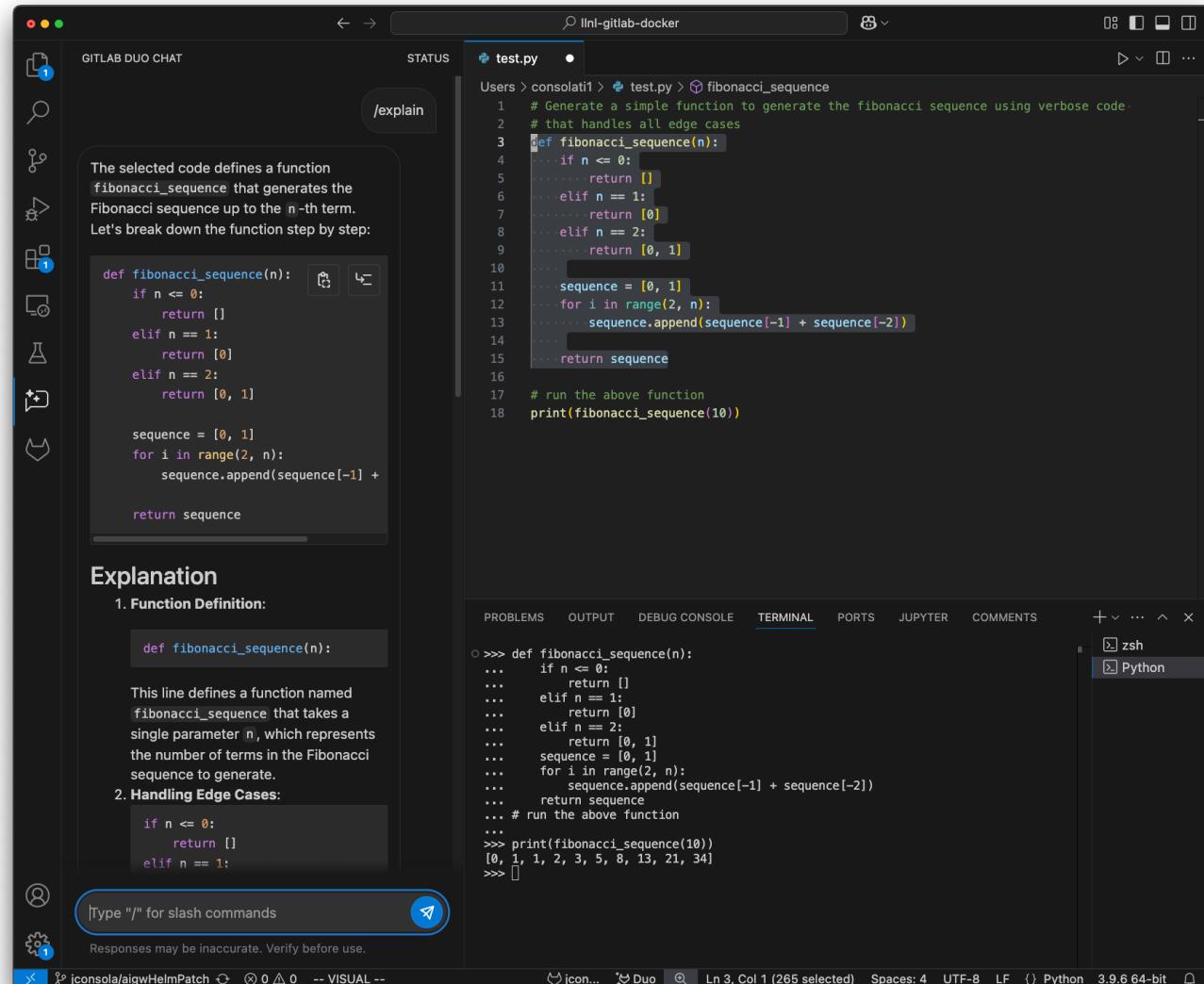
The screenshot shows a desktop IDE interface with the following elements:

- Top Bar:** Shows the title "llnl-gitlab-docker".
- Left Sidebar:** Contains icons for file operations (New, Open, Save, Find, Copy, Paste, Delete, Refresh), a search bar, and a "GITLAB DUO CHAT" section.
- Middle Area:** A code editor window titled "test.py" showing Python code for generating a Fibonacci sequence. The code is as follows:

```
1 # Generate a simple function to generate the fibonacci sequence using verbose code.
2 # that handles all edge cases
3 def fibonacci_sequence(n):
4     if n <= 0:
5         return []
6     elif n == 1:
7         return [0]
8     elif n == 2:
9         return [0, 1]
10    sequence = [0, 1]
11    for i in range(2, n):
12        sequence.append(sequence[-1] + sequence[-2])
13
14    return sequence
15
16
17 # run the above function
print(fibonacci_sequence(10))
```

- Bottom Bar:** Includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, JUPYTER, and COMMENTS. The TERMINAL tab is selected, showing a prompt "zsh" and a "Python" tab.
- GitLab Duo Chat Sidebar:** A floating window with the message: "I am GitLab Duo Chat, your personal AI-powered assistant. How can I help you today?". It lists several questions:
 - How do I change my password in GitLab?
 - How do I fork a project?
 - How do I clone a repository?
 - How do I create a template?A text input field at the bottom says "Type "/" for slash commands".
- Bottom Status Bar:** Shows the path "jconsole/aigwHelmPatch", file statistics ("0 0 0 0"), and the status "-- INSERT --". It also displays the current file type as "Python" and version "3.9.6 64-bit".

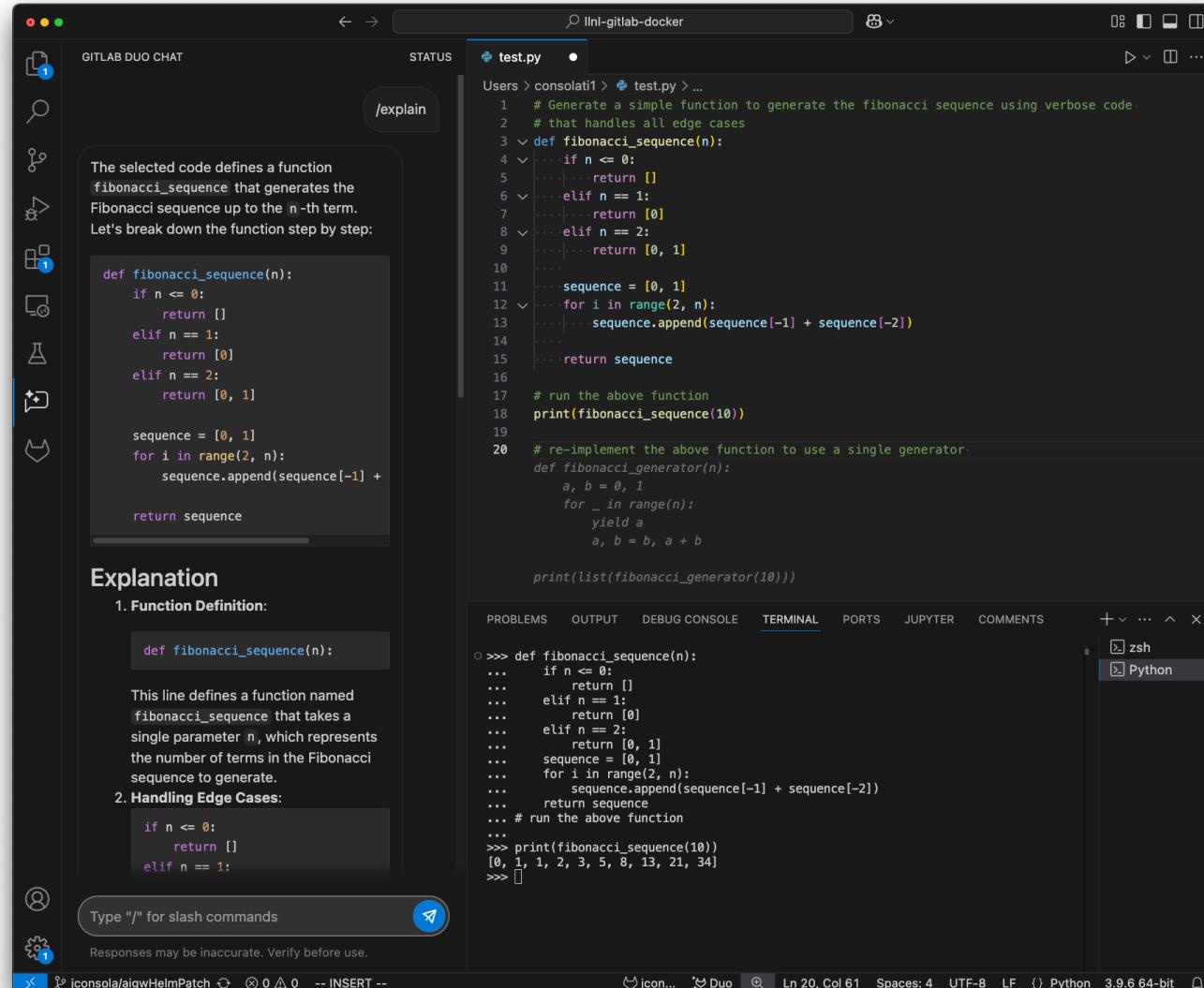
Desktop IDE – Example Workflow



The screenshot shows a desktop IDE interface with the following components:

- Code Editor:** The main window displays a Python file named `test.py`. The code defines a function `fibonacci_sequence` to generate the Fibonacci sequence up to the `n`-th term. The code uses a loop to build the sequence from `[0, 1]` up to `n`.
- Terminal:** Below the code editor, the terminal window shows the execution of the code. It runs the function with `n=10` and prints the resulting sequence: `[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]`.
- GitLab Duo Chat Panel:** A floating panel titled "GITLAB DUO CHAT" provides real-time feedback on the selected code. It states: "The selected code defines a function `fibonacci_sequence` that generates the Fibonacci sequence up to the `n`-th term. Let's break down the function step by step:" followed by the code snippet.
- Explanation Panel:** Below the GitLab panel, an "Explanation" section details the function definition and handling of edge cases. It highlights the first few lines of the function definition and the conditional logic for `n <= 0`, `n == 1`, and `n == 2`.
- Bottom Status Bar:** The status bar at the bottom shows the current workspace path (`jconsole/aigwHelmPatch`), the number of changes (`0 △ 0`), and other system information like "Ln 3, Col 1 (265 selected)" and "Spaces: 4".

Desktop IDE – Example Workflow



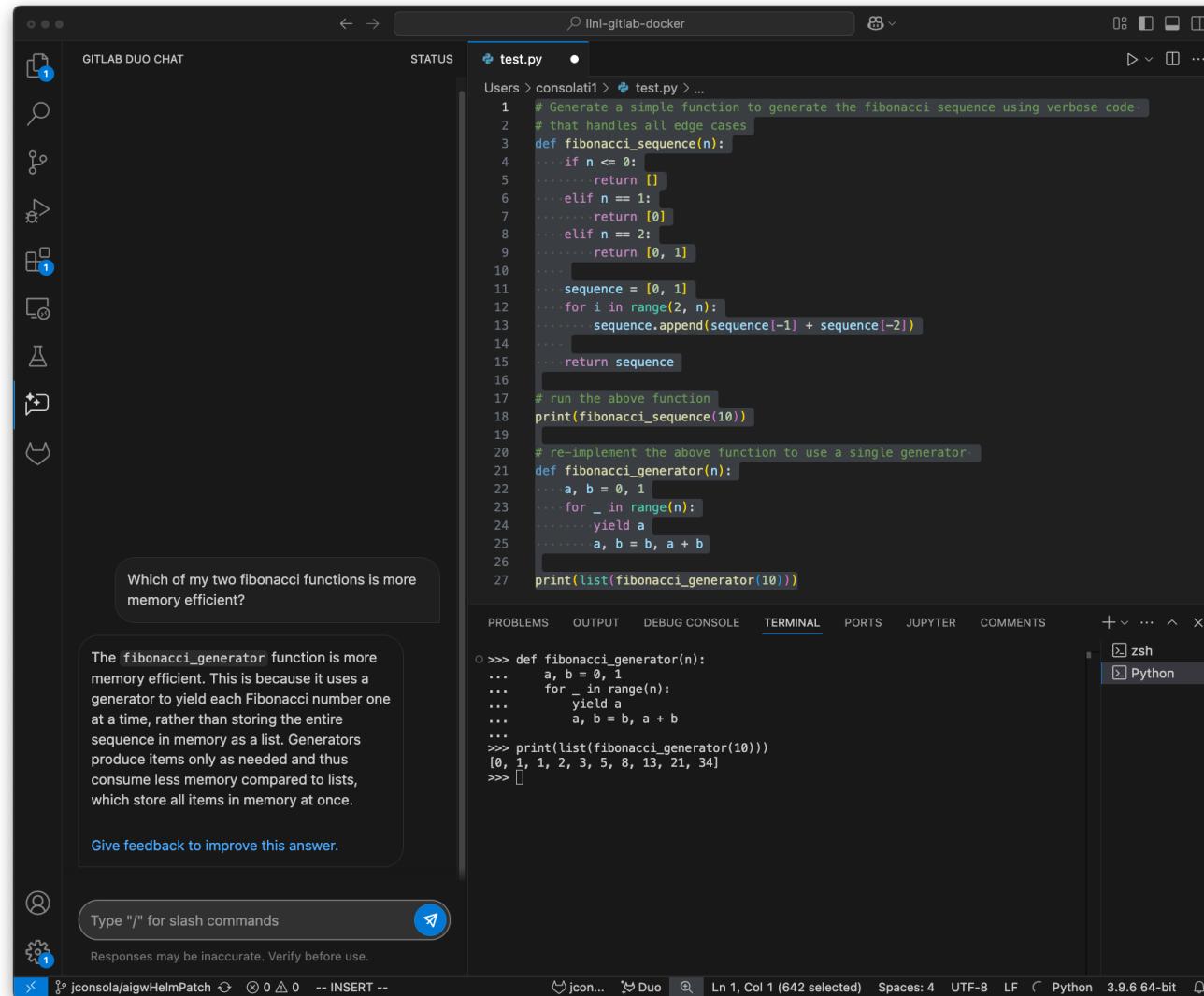
The screenshot shows a desktop IDE interface with a dark theme. The main area displays a Python file named `test.py` containing code to generate a Fibonacci sequence. A tooltip window is open over the code, providing an explanation of the function `fibonacci_sequence`. The explanation details the function's purpose, its step-by-step implementation, and highlights specific code snippets. Below the code editor, a terminal window is open, showing the execution of the code and its output. The terminal tab bar includes `PROBLEMS`, `OUTPUT`, `DEBUG CONSOLE`, `TERMINAL` (which is selected), `PORTS`, `JUPYTER`, and `COMMENTS`. The terminal output shows the generated Fibonacci sequence from 0 to 34.

```
test.py
1 # Generate a simple function to generate the fibonacci sequence using verbose code-
2 # that handles all edge cases
3 def fibonacci_sequence(n):
4     if n <= 0:
5         return []
6     elif n == 1:
7         return [0]
8     elif n == 2:
9         return [0, 1]
10    sequence = [0, 1]
11    for i in range(2, n):
12        sequence.append(sequence[-1] + sequence[-2])
13    return sequence
14
15    # run the above function
16    print(fibonacci_sequence(10))
17
18    # re-implement the above function to use a single generator
19    def fibonacci_generator(n):
20        a, b = 0, 1
21        for _ in range(n):
22            yield a
23            a, b = b, a + b
24
25    print(list(fibonacci_generator(10)))

```

```
>>> def fibonacci_sequence(n):
...     if n <= 0:
...         return []
...     elif n == 1:
...         return [0]
...     elif n == 2:
...         return [0, 1]
...     sequence = [0, 1]
...     for i in range(2, n):
...         sequence.append(sequence[-1] + sequence[-2])
...     return sequence
...
... # run the above function
... >>> print(fibonacci_sequence(10))
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
>>> 
```

Desktop IDE – Example Workflow



The screenshot shows a desktop IDE interface with a dark theme. On the left is a sidebar with various icons for file operations like copy, paste, search, and navigation. The main area has a title bar "llnl-gitlab-docker". The code editor window is titled "test.py" and contains Python code for generating Fibonacci sequences. The terminal window at the bottom shows a Python session where the generator function is tested.

```
# Generate a simple function to generate the fibonacci sequence using verbose code.
# that handles all edge cases
def fibonacci_sequence(n):
    if n <= 0:
        return []
    elif n == 1:
        return [0]
    elif n == 2:
        return [0, 1]
    sequence = [0, 1]
    for i in range(2, n):
        sequence.append(sequence[-1] + sequence[-2])
    return sequence

# run the above function
print(fibonacci_sequence(10))

# re-implement the above function to use a single generator
def fibonacci_generator(n):
    a, b = 0, 1
    for _ in range(n):
        yield a
        a, b = b, a + b

print(list(fibonacci_generator(10)))
```

The GitLab Duo Chat interface on the right shows a question about memory efficiency:

Which of my two fibonacci functions is more memory efficient?

The fibonacci_generator function is more memory efficient. This is because it uses a generator to yield each Fibonacci number one at a time, rather than storing the entire sequence in memory as a list. Generators produce items only as needed and thus consume less memory compared to lists, which store all items in memory at once.

Give feedback to improve this answer.

Type "/" for slash commands

Responses may be inaccurate. Verify before use.

Terminal output:

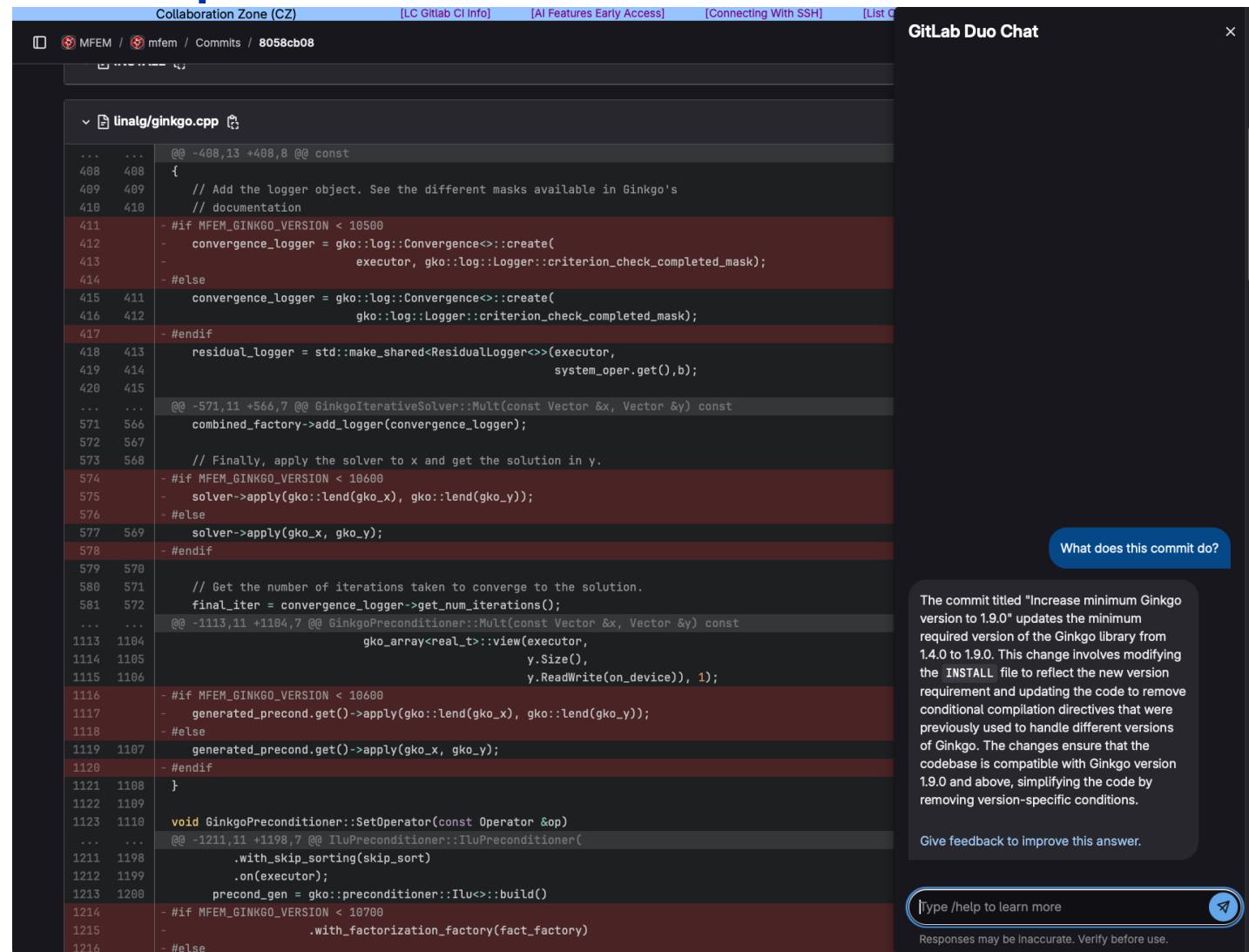
```
>>> def fibonacci_generator(n):
...     a, b = 0, 1
...     for _ in range(n):
...         yield a
...         a, b = b, a + b
...
>>> print(list(fibonacci_generator(10)))
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
>>> 
```



Web UI / IDE Integration

- Duo chat also available on the GitLab web UI with the same capabilities
 - Chat sessions will carry over between the web and your IDE
- The web IDE within GitLab (VSCode) also supports Duo in a similar fashion

Web UI - Example



The screenshot shows a GitLab Duo Chat window with a dark theme. On the left, a code editor displays a file named `linalg/ginkgo.cpp`. The code is a C++ file containing several conditional compilation blocks. A specific commit, `8058cb08`, is highlighted with a red background. The commit message is "Increase minimum Ginkgo version to 1.9.0". On the right, a callout box provides a detailed description of the commit's purpose:

The commit titled "Increase minimum Ginkgo version to 1.9.0" updates the minimum required version of the Ginkgo library from 1.4.0 to 1.9.0. This change involves modifying the `INSTALL` file to reflect the new version requirement and updating the code to remove conditional compilation directives that were previously used to handle different versions of Ginkgo. The changes ensure that the codebase is compatible with Ginkgo version 1.9.0 and above, simplifying the code by removing version-specific conditions.

Below the callout, there is a text input field with the placeholder "Type /help to learn more" and a "Give feedback to improve this answer" button.

```
@@ -408,13 +408,8 @@ const
408 408 {
409 409     // Add the logger object. See the different masks available in Ginkgo's
410 410     // documentation
411 411 - #if MFEM_GINKGO_VERSION < 10500
412 412 -     convergence_logger = gko::log::Convergence<>::create(
413 413         executor, gko::log::Logger::criterion_check_completed_mask);
414 414 - #else
415 411     convergence_logger = gko::log::Convergence<>::create(
416 412         gko::log::Logger::criterion_check_completed_mask);
417 417 - #endif
418 413     residual_logger = std::make_shared<ResidualLogger>(executor,
419 414         system_oper.get(), b);
420 415 ...
421 416 ...
422 417 ...
423 418 ...
424 419 ...
425 420 ...
426 421 ...
427 422 ...
428 423 ...
429 424 ...
430 425 ...
431 426 ...
432 427 ...
433 428 ...
434 429 ...
435 430 ...
436 431 ...
437 432 ...
438 433 ...
439 434 ...
440 435 ...
441 436 ...
442 437 ...
443 438 ...
444 439 ...
445 440 ...
446 441 ...
447 442 ...
448 443 ...
449 444 ...
450 445 ...
451 446 ...
452 447 ...
453 448 ...
454 449 ...
455 450 ...
456 451 ...
457 452 ...
458 453 ...
459 454 ...
460 455 ...
461 456 ...
462 457 ...
463 458 ...
464 459 ...
465 460 ...
466 461 ...
467 462 ...
468 463 ...
469 464 ...
470 465 ...
471 466 ...
472 467 ...
473 468 ...
474 469 ...
475 470 ...
476 471 ...
477 472 ...
478 473 ...
479 474 ...
480 475 ...
481 476 ...
482 477 ...
483 478 ...
484 479 ...
485 480 ...
486 481 ...
487 482 ...
488 483 ...
489 484 ...
490 485 ...
491 486 ...
492 487 ...
493 488 ...
494 489 ...
495 490 ...
496 491 ...
497 492 ...
498 493 ...
499 494 ...
500 495 ...
501 496 ...
502 497 ...
503 498 ...
504 499 ...
505 500 ...
506 501 ...
507 502 ...
508 503 ...
509 504 ...
510 505 ...
511 506 ...
512 507 ...
513 508 ...
514 509 ...
515 510 ...
516 511 ...
517 512 ...
518 513 ...
519 514 ...
520 515 ...
521 516 ...
522 517 ...
523 518 ...
524 519 ...
525 520 ...
526 521 ...
527 522 ...
528 523 ...
529 524 ...
530 525 ...
531 526 ...
532 527 ...
533 528 ...
534 529 ...
535 530 ...
536 531 ...
537 532 ...
538 533 ...
539 534 ...
540 535 ...
541 536 ...
542 537 ...
543 538 ...
544 539 ...
545 540 ...
546 541 ...
547 542 ...
548 543 ...
549 544 ...
550 545 ...
551 546 ...
552 547 ...
553 548 ...
554 549 ...
555 550 ...
556 551 ...
557 552 ...
558 553 ...
559 554 ...
560 555 ...
561 556 ...
562 557 ...
563 558 ...
564 559 ...
565 560 ...
566 561 ...
567 562 ...
568 563 ...
569 564 ...
570 565 ...
571 566 ...
572 567 ...
573 568 ...
574 569 ...
575 570 ...
576 571 ...
577 572 ...
578 573 ...
579 574 ...
580 575 ...
581 576 ...
582 577 ...
583 578 ...
584 579 ...
585 580 ...
586 581 ...
587 582 ...
588 583 ...
589 584 ...
590 585 ...
591 586 ...
592 587 ...
593 588 ...
594 589 ...
595 590 ...
596 591 ...
597 592 ...
598 593 ...
599 594 ...
600 595 ...
601 596 ...
602 597 ...
603 598 ...
604 599 ...
605 600 ...
606 601 ...
607 602 ...
608 603 ...
609 604 ...
610 605 ...
611 606 ...
612 607 ...
613 608 ...
614 609 ...
615 610 ...
616 611 ...
617 612 ...
618 613 ...
619 614 ...
620 615 ...
621 616 ...
622 617 ...
623 618 ...
624 619 ...
625 620 ...
626 621 ...
627 622 ...
628 623 ...
629 624 ...
630 625 ...
631 626 ...
632 627 ...
633 628 ...
634 629 ...
635 630 ...
636 631 ...
637 632 ...
638 633 ...
639 634 ...
640 635 ...
641 636 ...
642 637 ...
643 638 ...
644 639 ...
645 640 ...
646 641 ...
647 642 ...
648 643 ...
649 644 ...
650 645 ...
651 646 ...
652 647 ...
653 648 ...
654 649 ...
655 650 ...
656 651 ...
657 652 ...
658 653 ...
659 654 ...
660 655 ...
661 656 ...
662 657 ...
663 658 ...
664 659 ...
665 660 ...
666 661 ...
667 662 ...
668 663 ...
669 664 ...
670 665 ...
671 666 ...
672 667 ...
673 668 ...
674 669 ...
675 670 ...
676 671 ...
677 672 ...
678 673 ...
679 674 ...
680 675 ...
681 676 ...
682 677 ...
683 678 ...
684 679 ...
685 680 ...
686 681 ...
687 682 ...
688 683 ...
689 684 ...
690 685 ...
691 686 ...
692 687 ...
693 688 ...
694 689 ...
695 690 ...
696 691 ...
697 692 ...
698 693 ...
699 694 ...
700 695 ...
701 696 ...
702 697 ...
703 698 ...
704 699 ...
705 700 ...
706 701 ...
707 702 ...
708 703 ...
709 704 ...
710 705 ...
711 706 ...
712 707 ...
713 708 ...
714 709 ...
715 710 ...
716 711 ...
717 712 ...
718 713 ...
719 714 ...
720 715 ...
721 716 ...
722 717 ...
723 718 ...
724 719 ...
725 720 ...
726 721 ...
727 722 ...
728 723 ...
729 724 ...
730 725 ...
731 726 ...
732 727 ...
733 728 ...
734 729 ...
735 730 ...
736 731 ...
737 732 ...
738 733 ...
739 734 ...
740 735 ...
741 736 ...
742 737 ...
743 738 ...
744 739 ...
745 740 ...
746 741 ...
747 742 ...
748 743 ...
749 744 ...
750 745 ...
751 746 ...
752 747 ...
753 748 ...
754 749 ...
755 750 ...
756 751 ...
757 752 ...
758 753 ...
759 754 ...
760 755 ...
761 756 ...
762 757 ...
763 758 ...
764 759 ...
765 760 ...
766 761 ...
767 762 ...
768 763 ...
769 764 ...
770 765 ...
771 766 ...
772 767 ...
773 768 ...
774 769 ...
775 770 ...
776 771 ...
777 772 ...
778 773 ...
779 774 ...
780 775 ...
781 776 ...
782 777 ...
783 778 ...
784 779 ...
785 780 ...
786 781 ...
787 782 ...
788 783 ...
789 784 ...
790 785 ...
791 786 ...
792 787 ...
793 788 ...
794 789 ...
795 790 ...
796 791 ...
797 792 ...
798 793 ...
799 794 ...
800 795 ...
801 796 ...
802 797 ...
803 798 ...
804 799 ...
805 800 ...
806 801 ...
807 802 ...
808 803 ...
809 804 ...
810 805 ...
811 806 ...
812 807 ...
813 808 ...
814 809 ...
815 810 ...
816 811 ...
817 812 ...
818 813 ...
819 814 ...
820 815 ...
821 816 ...
822 817 ...
823 818 ...
824 819 ...
825 820 ...
826 821 ...
827 822 ...
828 823 ...
829 824 ...
830 825 ...
831 826 ...
832 827 ...
833 828 ...
834 829 ...
835 830 ...
836 831 ...
837 832 ...
838 833 ...
839 834 ...
840 835 ...
841 836 ...
842 837 ...
843 838 ...
844 839 ...
845 840 ...
846 841 ...
847 842 ...
848 843 ...
849 844 ...
850 845 ...
851 846 ...
852 847 ...
853 848 ...
854 849 ...
855 850 ...
856 851 ...
857 852 ...
858 853 ...
859 854 ...
860 855 ...
861 856 ...
862 857 ...
863 858 ...
864 859 ...
865 860 ...
866 861 ...
867 862 ...
868 863 ...
869 864 ...
870 865 ...
871 866 ...
872 867 ...
873 868 ...
874 869 ...
875 870 ...
876 871 ...
877 872 ...
878 873 ...
879 874 ...
880 875 ...
881 876 ...
882 877 ...
883 878 ...
884 879 ...
885 880 ...
886 881 ...
887 882 ...
888 883 ...
889 884 ...
890 885 ...
891 886 ...
892 887 ...
893 888 ...
894 889 ...
895 890 ...
896 891 ...
897 892 ...
898 893 ...
899 894 ...
900 895 ...
901 896 ...
902 897 ...
903 898 ...
904 899 ...
905 900 ...
906 901 ...
907 902 ...
908 903 ...
909 904 ...
910 905 ...
911 906 ...
912 907 ...
913 908 ...
914 909 ...
915 910 ...
916 911 ...
917 912 ...
918 913 ...
919 914 ...
920 915 ...
921 916 ...
922 917 ...
923 918 ...
924 919 ...
925 920 ...
926 921 ...
927 922 ...
928 923 ...
929 924 ...
930 925 ...
931 926 ...
932 927 ...
933 928 ...
934 929 ...
935 930 ...
936 931 ...
937 932 ...
938 933 ...
939 934 ...
940 935 ...
941 936 ...
942 937 ...
943 938 ...
944 939 ...
945 940 ...
946 941 ...
947 942 ...
948 943 ...
949 944 ...
950 945 ...
951 946 ...
952 947 ...
953 948 ...
954 949 ...
955 950 ...
956 951 ...
957 952 ...
958 953 ...
959 954 ...
960 955 ...
961 956 ...
962 957 ...
963 958 ...
964 959 ...
965 960 ...
966 961 ...
967 962 ...
968 963 ...
969 964 ...
970 965 ...
971 966 ...
972 967 ...
973 968 ...
974 969 ...
975 970 ...
976 971 ...
977 972 ...
978 973 ...
979 974 ...
980 975 ...
981 976 ...
982 977 ...
983 978 ...
984 979 ...
985 980 ...
986 981 ...
987 982 ...
988 983 ...
989 984 ...
990 985 ...
991 986 ...
992 987 ...
993 988 ...
994 989 ...
995 990 ...
996 991 ...
997 992 ...
998 993 ...
999 994 ...
1000 995 ...
1001 996 ...
1002 997 ...
1003 998 ...
1004 999 ...
1005 1000 ...
1006 1001 ...
1007 1002 ...
1008 1003 ...
1009 1004 ...
1010 1005 ...
1011 1006 ...
1012 1007 ...
1013 1008 ...
1014 1009 ...
1015 1010 ...
1016 1011 ...
1017 1012 ...
1018 1013 ...
1019 1014 ...
1020 1015 ...
1021 1016 ...
1022 1017 ...
1023 1018 ...
1024 1019 ...
1025 1020 ...
1026 1021 ...
1027 1022 ...
1028 1023 ...
1029 1024 ...
1030 1025 ...
1031 1026 ...
1032 1027 ...
1033 1028 ...
1034 1029 ...
1035 1030 ...
1036 1031 ...
1037 1032 ...
1038 1033 ...
1039 1034 ...
1040 1035 ...
1041 1036 ...
1042 1037 ...
1043 1038 ...
1044 1039 ...
1045 1040 ...
1046 1041 ...
1047 1042 ...
1048 1043 ...
1049 1044 ...
1050 1045 ...
1051 1046 ...
1052 1047 ...
1053 1048 ...
1054 1049 ...
1055 1050 ...
1056 1051 ...
1057 1052 ...
1058 1053 ...
1059 1054 ...
1060 1055 ...
1061 1056 ...
1062 1057 ...
1063 1058 ...
1064 1059 ...
1065 1060 ...
1066 1061 ...
1067 1062 ...
1068 1063 ...
1069 1064 ...
1070 1065 ...
1071 1066 ...
1072 1067 ...
1073 1068 ...
1074 1069 ...
1075 1070 ...
1076 1071 ...
1077 1072 ...
1078 1073 ...
1079 1074 ...
1080 1075 ...
1081 1076 ...
1082 1077 ...
1083 1078 ...
1084 1079 ...
1085 1080 ...
1086 1081 ...
1087 1082 ...
1088 1083 ...
1089 1084 ...
1090 1085 ...
1091 1086 ...
1092 1087 ...
1093 1088 ...
1094 1089 ...
1095 1090 ...
1096 1091 ...
1097 1092 ...
1098 1093 ...
1099 1094 ...
1100 1095 ...
1101 1096 ...
1102 1097 ...
1103 1098 ...
1104 1099 ...
1105 1100 ...
1106 1101 ...
1107 1102 ...
1108 1103 ...
1109 1104 ...
1110 1105 ...
1111 1106 ...
1112 1107 ...
1113 1108 ...
1114 1109 ...
1115 1110 ...
1116 1111 ...
1117 1112 ...
1118 1113 ...
1119 1114 ...
1120 1115 ...
1121 1116 ...
1122 1117 ...
1123 1118 ...
1124 1119 ...
1125 1120 ...
1126 1121 ...
1127 1122 ...
1128 1123 ...
1129 1124 ...
1130 1125 ...
1131 1126 ...
1132 1127 ...
1133 1128 ...
1134 1129 ...
1135 1130 ...
1136 1131 ...
1137 1132 ...
1138 1133 ...
1139 1134 ...
1140 1135 ...
1141 1136 ...
1142 1137 ...
1143 1138 ...
1144 1139 ...
1145 1140 ...
1146 1141 ...
1147 1142 ...
1148 1143 ...
1149 1144 ...
1150 1145 ...
1151 1146 ...
1152 1147 ...
1153 1148 ...
1154 1149 ...
1155 1150 ...
1156 1151 ...
1157 1152 ...
1158 1153 ...
1159 1154 ...
1160 1155 ...
1161 1156 ...
1162 1157 ...
1163 1158 ...
1164 1159 ...
1165 1160 ...
1166 1161 ...
1167 1162 ...
1168 1163 ...
1169 1164 ...
1170 1165 ...
1171 1166 ...
1172 1167 ...
1173 1168 ...
1174 1169 ...
1175 1170 ...
1176 1171 ...
1177 1172 ...
1178 1173 ...
1179 1174 ...
1180 1175 ...
1181 1176 ...
1182 1177 ...
1183 1178 ...
1184 1179 ...
1185 1180 ...
1186 1181 ...
1187 1182 ...
1188 1183 ...
1189 1184 ...
1190 1185 ...
1191 1186 ...
1192 1187 ...
1193 1188 ...
1194 1189 ...
1195 1190 ...
1196 1191 ...
1197 1192 ...
1198 1193 ...
1199 1194 ...
1200 1195 ...
1201 1196 ...
1202 1197 ...
1203 1198 ...
1204 1199 ...
1205 1200 ...
1206 1201 ...
1207 1202 ...
1208 1203 ...
1209 1204 ...
1210 1205 ...
1211 1206 ...
1212 1207 ...
1213 1208 ...
1214 1209 ...
1215 1210 ...
1216 1211 ...
1217 1212 ...
1218 1213 ...
1219 1214 ...
1220 1215 ...
1221 1216 ...
1222 1217 ...
1223 1218 ...
1224 1219 ...
1225 1220 ...
1226 1221 ...
1227 1222 ...
1228 1223 ...
1229 1224 ...
1230 1225 ...
1231 1226 ...
1232 1227 ...
1233 1228 ...
1234 1229 ...
1235 1230 ...
1236 1231 ...
1237 1232 ...
1238 1233 ...
1239 1234 ...
1240 1235 ...
1241 1236 ...
1242 1237 ...
1243 1238 ...
1244 1239 ...
1245 1240 ...
1246 1241 ...
1247 1242 ...
1248 1243 ...
1249 1244 ...
1250 1245 ...
1251 1246 ...
1252 1247 ...
1253 1248 ...
1254 1249 ...
1255 1250 ...
1256 1251 ...
1257 1252 ...
1258 1253 ...
1259 1254 ...
1260 1255 ...
1261 1256 ...
1262 1257 ...
1263 1258 ...
1264 1259 ...
1265 1260 ...
1266 1261 ...
1267 1262 ...
1268 1263 ...
1269 1264 ...
1270 1265 ...
1271 1266 ...
1272 1267 ...
1273 1268 ...
1274 1269 ...
1275 1270 ...
1276 1271 ...
1277 1272 ...
1278 1273 ...
1279 1274 ...
1280 1275 ...
1281 1276 ...
1282 1277 ...
1283 1278 ...
1284 1279 ...
1285 1280 ...
1286 1281 ...
1287 1282 ...
1288 1283 ...
1289 1284 ...
1290 1285 ...
1291 1286 ...
1292 1287 ...
1293 1288 ...
1294 1289 ...
1295 1290 ...
1296 1291 ...
1297 1292 ...
1298 1293 ...
1299 1294 ...
1300 1295 ...
1301 1296 ...
1302 1297 ...
1303 1298 ...
1304 1299 ...
1305 1300 ...
1306 1301 ...
1307 1302 ...
1308 1303 ...
1309 1304 ...
1310 1305 ...
1311 1306 ...
1312 1307 ...
1313 1308 ...
1314 1309 ...
1315 1310 ...
1316 1311 ...
1317 1312 ...
1318 1313 ...
1319 1314 ...
1320 1315 ...
1321 1316 ...
1322 1317 ...
1323 1318 ...
1324 1319 ...
1325 1320 ...
1326 1321 ...
1327 1322 ...
1328 1323 ...
1329 1324 ...
1330 1325 ...
1331 1326 ...
1332 1327 ...
1333 1328 ...
1334 1329 ...
1335 1330 ...
1336 1331 ...
1337 1332 ...
1338 1333 ...
1339 1334 ...
1340 1335 ...
1341 1336 ...
1342 1337 ...
1343 1338 ...
1344 1339 ...
1345 1340 ...
1346 1341 ...
1347 1342 ...
1348 1343 ...
1349 1344 ...
1350 1345 ...
1351 1346 ...
1352 1347 ...
1353 1348 ...
1354 1349 ...
1355 1350 ...
1356 1351 ...
1357 1352 ...
1358 1353 ...
1359 1354 ...
1360 1355 ...
1361 1356 ...
1362 1357 ...
1363 1358 ...
1364 1359 ...
1365 1360 ...
1366 1361 ...
1367 1362 ...
1368 1363 ...
1369 1364 ...
1370 1365 ...
1371 1366 ...
1372 1367 ...
1373 1368 ...
1374 1369 ...
1375 1370 ...
1376 1371 ...
1377 1372 ...
1378 1373 ...
1379 1374 ...
1380 1375 ...
1381 1376 ...
1382 1377 ...
1383 1378 ...
1384 1379 ...
1385 1380 ...
1386 1381 ...
1387 1382 ...
1388 1383 ...
1389 1384 ...
1390 1385 ...
1391 1386 ...
1392 1387 ...
1393 1388 ...
1394 1389 ...
1395 1390 ...
1396 1391 ...
1397 1392 ...
1398 1393 ...
1399 1394 ...
1400 1395 ...
1401 1396 ...
1402 1397 ...
1403 1398 ...
1404 1399 ...
1405 1400 ...
1406 1401 ...
1407 1402 ...
1408 1403 ...
1409 1404 ...
1410 1405 ...
1411 1406 ...
1412 1407 ...
1413 1408 ...
1414 1409 ...
1415 1410 ...
1416 1411 ...
1417 1412 ...
1418 1413 ...
1419 1414 ...
1420 1415 ...
1421 1416 ...
1422 1417 ...
1423 1418 ...
1424 1419 ...
1425 1420 ...
1426 1421 ...
1427 1422 ...
1428 1423 ...
1429 1424 ...
1430 1425 ...
1431 1426 ...
1432 1427 ...
1433 1428 ...
1434 1429 ...
1435 1430 ...
1436 1431 ...
1437 1432 ...
1438 1433 ...
1439 1434 ...
1440 1435 ...
1441 1436 ...
1442 1437 ...
1443 1438 ...
1444 1439 ...
1445 1440 ...
1446 1441 ...
1447 1442 ...
1448 1443 ...
1449 1444 ...
1450 1445 ...
1451 1446 ...
1452 1447 ...
1453 1448 ...
1454 1449 ...
1455 1450 ...
1456 1451 ...
1457 1452 ...
1458 1453 ...
1459 1454 ...
1460 1455 ...
1461 1456 ...
1462 1457 ...
1463 1458 ...
1464 1459 ...
1465 1460 ...
1466 1461 ...
1467 1462 ...
1468 1463 ...
1469 1464 ...
1470 1465 ...
1471 1466 ...
1472 1467 ...
1473 1468 ...
1474 1469 ...
1475 1470 ...
1476 1471 ...
1477 1472 ...
1478 1473 ...
1479 1474 ...
1480 1475 ...
1481 1476 ...
1482 1477 ...
1483 1478 ...
1484 1479 ...
1485 1480 ...
1486 1481 ...
1487 1482 ...
1488 1483 ...
1489 1484 ...
1490 1485 ...
1491 1486 ...
1492 1487 ...
1493 1488 ...
1494 1489 ...
1495 1490 ...
1496 1491 ...
1497 1492 ...
1498 1493 ...
1499 1494 ...
1500 1495 ...
1501 1496 ...
1502 1497 ...
1503 1498 ...
1504 1499 ...
1505 1500 ...
1506 1501 ...
1507 1502 ...
1508 1503 ...
1509 1504 ...
1510 1505 ...
1511 1506 ...
1512 1507 ...
1513 1508 ...
1514 1509 ...
1515 1510 ...
1516 1511 ...
1517 1512 ...
1518 1513 ...
1519 1514 ...
1520 1515 ...
1521 1516 ...
1522 1517 ...
1523 1518 ...
1524 1519 ...
1525 1520 ...
1526 1521 ...
1527 1522 ...
1528 1523 ...
1529 1524 ...
1530 1525 ...
1531 1526 ...
1532 1527 ...
1533 1528 ...
1534 1529 ...
1535 1530 ...
1536 1531 ...
1537 1532 ...
1538 1533 ...
1539 1534 ...
1540 1535 ...
1541 1536 ...
1542 1537 ...
1543 1538 ...
1544 1539 ...
1545 1540 ...
1546 1541 ...
1547 1542 ...
1548 1543 ...
1549 1544 ...
1550 1545 ...
1551 1546 ...
1552 1547 ...
1553 1548 ...
1554 1549 ...
1555 1550 ...
1556 1551 ...
1557 1552 ...
1558 1553 ...
1559 1554 ...
1560 1555 ...
1561 1556 ...
1562 1557 ...
1563 1558 ...
1564 1559 ...
1565 1560 ...
1566 1561 ...
1567 1562 ...
1568 1563 ...
1569 1564 ...
1570 1565 ...
1571 1566 ...
1572 1567 ...
1573 1568 ...
1574 1569 ...
1575 1570 ...
1576 1571 ...
1577 1572 ...
1578 1573 ...

```

Supported Models & Hosting Environments

Hosting Platform	GitLab Supported Models	Notes
Fully on-prem (\$\$\$)	<ul style="list-style-type: none">Mistral series Mixtral series CodeGemma (future) CodeLlama (future) DeepSeek Coder (future) 	<ul style="list-style-type: none">On-prem involves procuring new hardware or reconfiguring HPC nodesDeepSeek not LLNL approved
Amazon Bedrock (\$) (GovCloud instance)	<ul style="list-style-type: none">Mixtral series Claude 3.5 Sonnet 	<ul style="list-style-type: none">Usable in CZ only
Azure OpenAI (\$) (Government instance)	<ul style="list-style-type: none">GPT series Llama 3 series 	<ul style="list-style-type: none">Usable in CZ onlyPilot currently using GPT-4o

   Icons represent degree of accuracy in model responses for GitLab use cases

Combinatorial problem of model support, model accuracy, cost, security, and hardware availability



What's next?

- Launching pilot for RZ GitLab. Will initially run models hosted on rzvernal.
- Evaluate viability of the Mixtral 8x22B-it model
 - CZ pilot previously ran Mistral 7B-it which was insufficient
 - Expecting results better than Mistral 7B but worse than GPT. Good enough?
- Ensure stable infrastructure prior to general availability launch
- Feature parity of Duo across different instances



Seeking Feedback

- Accuracy & speed of responses
- Ease of providing context to the model
- Does it enable you to do things you couldn't previously? What are the time savings?
- How can we make this better?
- If you use both CZ & RZ, do you notice differences between GPT and Mixtral responses?
- lc-gitlab-admin@llnl.gov

