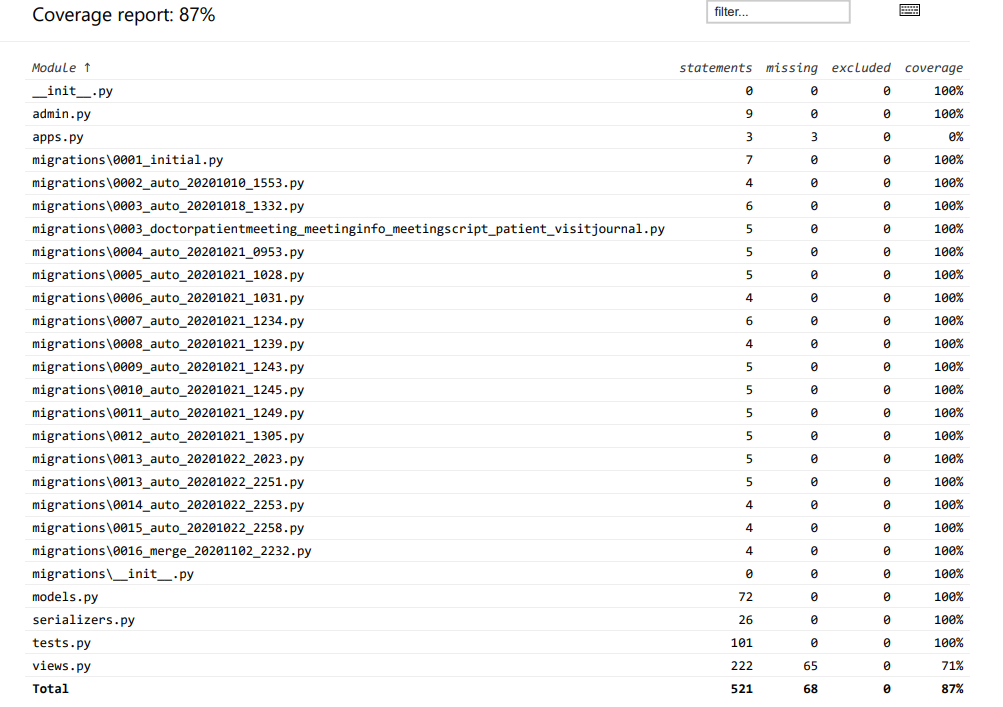
Synchronously, during class, your team will have ***12 minutes*** to present a demo as well as team and individual reflections from the semester:

* **Demo**. a working, ***deployed***, version of the app, that shows the ***complete set***of final client facing features
  + Features to show:
    - Log in
    - Sign up
      * Error messages (with empty sign up, with wrong passwords, error messages on login screen with non-existing user info and wrong password popups)
    - Record workflow with existing doctors
    - Record workflow with new doctor & facility
    - View timeline visit journal
    - View user profile
      * Doctor and view doctor card
      * Facility and view facility card
    - Log out
* **License Choice**. explain which license your team and client ***agreed to*** for the app and why it was chosen
  + We chose to use a closed-source, proprietary license for our app, because our client is looking to pitch this app to investors and potentially turn it into a startup. For that reason, we decided to choose an **Attribution-NonCommercial-NoDerivs** license.
  + This means that the code for Cornelius cannot be modified or used for commercial purposes, and if it is used by any other party they must give credit to the authors.
  + Because there are competitors working on similar products, it is important that we establish that our code cannot be repurposed or adapted for different apps.
* **User Testing Results**. one positive ***and*** one critical thing learned from user testing your app with potential users (including user quotes is encouraged).   
  Share one thing ***each person*** learned during the semester from interacting with potential users.
  + One positive:
    - very clean UI, liked how the add appointment is the main focus on navigation”
    - “Easy to use, can view timeline easily and record and get very accurate transcript. I love it!”
  + One critical:
    - Add an \* for required fields when adding appointment information
    - Less words on recording screen and bigger button
  + Melissa: I really learned from observing how much more complicated it is for users to actually go through app functionalities when it is unknown. I already was a big supporter of minimalistic design and intuitive design but through user testing I realized how important it actually is.
  + Kiori: I learned the importance of getting frequent external user feedback because when you are working on the app for so long, the workflow becomes intuitive to the developers and we tend to miss bugs or overestimate the usability of the app.
  + Matt: I learned the importance of onboarding for an app. I usually skip through them and try to figure it out as I go, but it really is important to guide first-time users through the purpose and features of an app. Because I had been working on Cornelius for so long, I just assumed everyone else would know how it worked.
  + Siyi: I learned that it is very different for users to use the app, compared to developers. Some details & bugs we tend to ignore because we are so familiar with the workflow. So it is very useful to have user testing so that we know how to improve the usability of the app.
* **Testing Results**. demo the results from running your automated tests, ***showing Code Coverage results***, and describe one manual test you do regularly as well as results from an Accessibility checking tool.   
  Share one thing ***each person***learned or changed by testing their code regularly.

Code coverage, automated test, manually testing our endpoints



* + Melissa: I learned to make my code cleaner and effectively do error catching in the front end. That really helped when I was going through manual testing to understand what was going on and how to fix it.
  + Kiori: I learned how to
  + Matt: Testing my code kept me more accountable because I had to think of more edge cases when I was both writing code and writing tests. It also made me think about the key features of the code and what was most important to test.
  + Siyi: I learned the importance of automation testing. I have never implemented automation testing before. Before having automation testing, I test all the endpoints manually and it’s very time-consuming; With the automation testing, it saves me a lot of time.
* **App Review**. ***contrast*** the completed app compared to where you planned it to be in your initial Wireframe and Project Plan and how you have worked to make the app as inclusive as possible and reduce potential negative outcomes.   
  Share one non-technical thing ***each person*** learned during the semester by working on your app or with your client.
  + One thing that was different was the information we required for signing up, for instance we eliminated phone numbers, and date of birth because we did not really need that information and wanted to have justified data needs. Some other features that we were supposed to include but did not were the forgot password functionality, the FAQ page, and the search in timeline view functionality. One major change was in the way the timeline view looked, in our completed app we made each visit a touchable button to help declutter that interface and make it easier to differentiate and see one visit to another. This really helped make our app more inclusive because it made the interface easier to navigate. Lastly, we also eliminated the user insurance information and address within the app to reduce unnecessary data we have from our users and thus reduce the impact of potential data breaches. We also changed the order of setting up a recording so that it would require less work on the spot for our users.
  + Melissa: I learned that when you are working with clients who are non-technical it is very important to not only have initiative to decide most of the technical issues but also to be able to explain these in non technical terms. This is a very important skill that I saw myself improving at more and more towards the end of the semester.
  + Kiori: I learned the importance of being explicit in what features can be achieved and what the expectations are from both sides off the bat. This can prevent either side from being disappointed and ensure that the developers are able to prioritize the most important features to the client within the time frame given.
  + Matt: I learned how important it is to consistently get client feedback. Sometimes we would head in a direction that made sense to us but did not meet the client’s expectations. I realized we were working for a stakeholder at the end of the day, which separated this from a normal class project where we were our own client.
  + Siyi: I learned the importance of communicating technical terms with non-technical clients. It is important to tell the client the difficulty of the feature and how long we will take and be sure both sides have the same expectation because non-technical persons tend to underestimate the difficulties of the task.
* **Timeline**. show ***your Burndown chart*** and provide a brief timeline of significant events that occurred this Sprint and how communication was handled for each event (i.e., how each person was involved or learned about it later)  
  Share one thing ***each person*** learned or changed by trying to estimate and plan their work using tools, such as Issues and Milestones.
  + As it was our last sprint, when we talked with our client at the beginning of last week, we really tried to explicitly discuss what features were most important for them to have by the end of this semester. Since we completed the main functionality that they requested for our MVP, this sprint was dedicated to enhancing usability and experience through small front end changes such as including loading screens to indicate fetching data so they know that the app is not frozen. The upticks in the chart represent small additional changes to fix bugs encountered throughout development. Overall, our team has become better at pacing ourselves with the issues and were able to complete all of the tasks that we promised to deliver to our clients.
  + Melissa: I found that something that really helped me plan my time was to use milestones and very simple issues that were only 2-3 points. This made me more effective because I was able to code for the time I needed and not spend too much time. Also being able to view my progress and compare it to my previous activity made me want to be better each new sprint.
  + Kiori: At the beginning of the semester, we always had problems of features being dependent on each other, which caused some members to just be on stand-by for up to a few days, but I think we got better at making sure that we were all being productive throughout the sprint by working incrementally/equally rather than all at once.
  + Matt: I learned how to gauge how much work to give myself in a given sprint. Since each task was projected to take up to 4 hours, I knew that I shouldn’t give myself more than I could realistically handle. By the end of this semester, I felt much less stressed because I stopped overloading myself and started to learn my limits.
  + Siyi: I learned how to break down the features. At the beginning, we have very general issues that take a couple days to close. Then we realize we can break down to more small tasks and each with small weight. In this way, we are more clear of what should we do and better keep track of the progress
* **Retrospective**. reflect on what has worked to make your team successful ***and*** what could still be improved: highlight changes made to your Team Contract and how those did (or did not) help foster teamwork; and what ***concrete experiments*** you took to attempt to improve and how they helped (or not) your team's communication and velocity.  
  Share one thing ***each person*** learned or changed by using the Agile process to manage the coding process.
  + Success: I think one thing that we experimented with is making our Issues more granular. At the beginning of the semester, we would make one issue for whole features. However, we changed this towards the second sprint by including more specific tasks (transcription functionality vs. set up meeting script endpoint / create MeetingScriptModel) which helped our team’s velocity because we were able to track each other’s progress on a feature more accurately and help us prioritize / distribute our issues at the beginning of each sprint.
  + Improvements: One thing we could improve upon is our branch workflow. We had a consistent method in terms of always merging to development before production, but we only started creating a branch for each milestone so we can pinpoint our code on each sprint.
  + Melissa: One thing that really changed the way I code was to divide my sprints using the sprint estimation point scales. Before I would have to sit down and code for 9-10hrs straight because I underestimated each thing. At the end I started managing my time better.
  + Kiori: One thing that really worked for our team was peer programming particularly during processes that required code integration. Since we all worked on different parts of the codebase, it was more efficient to collaborate in real time rather than trying to figure out what part of the code is doing or waiting for a message response.
  + Matt: Communication was a large part of what made our team successful in quickly addressing bugs or incorporating feedback. In our team contract, we stated that we would respond to messages promptly which has been true throughout the project. I’ve learned a lot about how to succeed in a remote team because of this experience.
  + Siyi: I learned the importance of retrospective. After every sprint, we realize we have some issues and we fixed those through retrospective. It makes the team more effective and avoids repeating the same mistake we made in the last sprint .
* **Team Review**. most important way your team has improved during the semester ***and*** one thing that could still be improved.   
  Share one thing ***each person*** learned during the semester about creating a positive team culture.
  + Most important way we have improved: one thing that really helped us was that we started having more constant check ins. This helped us understand how we were progressing and what were some things we needed help with. Doing these more earlier in the sprint helped us put pressure on each other and thus get started with the work earlier instead of leaving it until the end.
  + Could still improve: One thing that we really struggled with was organizing our schedules with the time difference. This caused us to sometimes have to rush to complete some of the documentation during the week and fall behind on some of the key functionality. Also at times that caused us to struggle to split up the work evenly.
  + Melissa: One thing that I learned was that it is important to voice your concerns early and openly. Sometimes we would have to do more work one sprint vs another sprint because of everybody’s schedule and when there situations arose it was important to be upfront and avoid keeping thoughts to yourself. By being open we were able to navigate team conflicts and create an inclusive team culture.
  + Kiori: One thing I learned was the importance of stopping to acknowledge and appreciate each other’s efforts. Especially since we were all remote, it was easy to feel disconnected with each other, but positive feedback was important to keep up team spirit.
  + Matt: Being remote definitely impacted our team’s culture from my perspective. Because we would only interact over Zoom or over text, it was hard to feel connected to the team. I learned that it’s important to ask about teammates’ weeks or have a conversation that isn’t completely focused on the app we were building.
  + Siyi: I think one thing I learn when working remotely is to work together in Zoom meetings. So we all teammates can have instant feedback and get help from others. Otherwise, it is easy to feel disconnected or don’t know what the teammates are doing.