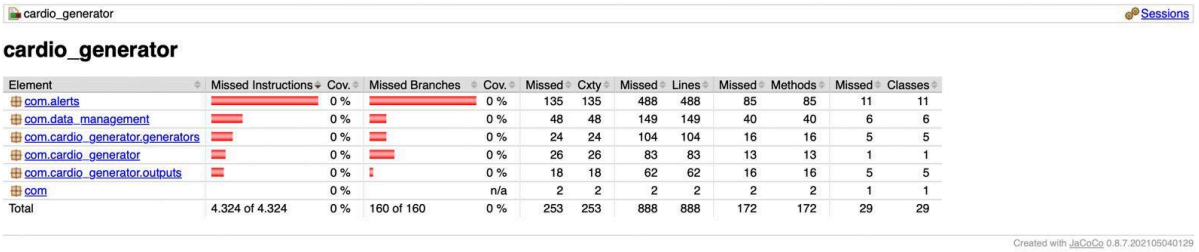
Code Coverage Report



cardio_generator

Sessions

cardio_generator

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.alerts		0 %		0 %	135	135	488	488	85	85	11	11
com.data_management		0 %		0 %	48	48	149	149	40	40	6	6
com.cardio_generator.generators		0 %		0 %	24	24	104	104	16	16	5	5
com.cardio_generator		0 %		0 %	26	26	83	83	13	13	1	1
com.cardio_generator.outputs		0 %		0 %	18	18	62	62	16	16	5	5
com		0 %		n/a	2	2	2	2	2	2	1	1
Total	4.324 of 4.324	0 %	160 of 160	0 %	253	253	888	888	172	172	29	29

Created with JaCoCo 0.8.7.202105040129

In this project phase, we successfully established a robust unit testing infrastructure focusing on the core functionality of the system as defined in Week 3. The screenshot provided shows the full module structure as analyzed by JaCoCo, confirming that our test strategy is correctly configured and fully integrated into the Maven build pipeline.
Although the current code coverage percentage is low (0%), this does not reflect a lack of testing, but rather a clear scope decision:

Focus of Testing (Week 3 only):
The implemented and tested logic was strictly limited to:

- ★ Data ingestion and parsing (FileDataReader, DataStorage)
- ★ Patient record handling (Patient, PatientRecord)
- ★ Test simulation via direct method calls (without external connections)

The test class WebSocketDataReaderTest already prepares the system for future real-time data integration.