|  |  |
| --- | --- |
| MUST | * Set requirements on Kanban board * Have a complete overhaul of the existing website front-end * Have a full build of the application * Create test suites for the application   + unit, integration, user-acceptance, non-functional (using JMeter) * Use static analysis tool (SonarQube) while building * Have sign-up functionality for multiple user accounts * Front-end style consistency * users can CRUD   + albums   + artists   + tracks   + genres * have the ability to browse (but not CRUD) without login * user can CRUD as many playlist as they like on their home screen |
| SHOULD | * Aim to reach industry standard of 80% test coverage * significantly augment the existing website back-end * Use SonarQube to resolve all code smells * Create a wireframe diagram for the front-end * Have the ability to search   + by track   + by album   + by artist * users can see cards for each playlist on their home screen * each album has its own page   + contains a list of songs   + contains a link to the artist page   + each track links to the track page * each artist has their own page   + contains a list of albums   + each album links to an album page * each track has its own page   + showing name, lyrics ang genre   + contains links to album and artist * each genre has its own page   + contains a list of tracks   + each track links to its track page * user should CRUD playlist by song id, name or genre * response times should be <10 ms * latency should be <2 seconds at 10,000 concurrent users * throughput rate should be >300/s * RAM allocation should be minimal with few (if any) memory leaks * should be spike, load, stress and soak tested |
| COULD | * Link Kanban board to git repo * Utilise versioning to create different releases of the application |
| WON’T | * have album art * have artist image * have audio files |