Minimum Viable Product Plan Vision & Justifications

# Product Vision

*Instructions: Create a vision for the product. The vision should include the intended target market or population and the outcome you are attempting to achieve.*

|  |  |
| --- | --- |
| **Product Vision** | Our mission to create a cross-platform application that tracks the virus and the status of virus test kits in Africa and the Middle East. That will help people get treatment where it’s needed the most, saving one soul is like saving all the human souls … |

# Product Roadmap Justifications

*Instructions: After creating the product roadmap, choose 1 feature in each phase and provide a simple justification for placing that feature into the phase that you put it in.*

|  |  |  |
| --- | --- | --- |
| **Phase** | **Feature** | **Justification** |
| 1 | Initial Architecture/Database Setup & Design | It’s logical to start with this because the initial architecture or the database setup and design can be considered as the main stepping stone. None of the data storage functionality will work without the base architecture or database setup. |
| 2 | Security Update | Security is a must for sure, no one can argue that by the time phase 1 is finished and we have an up and running web app. The second phase seems to be a good phase to launch security updates by considering all the vulnerabilities. |
| 3 | Develop iOS Compatible Mobile App  Develop android Compatible Mobile App | Developing iOS or android application are important but I think running web portal is more essential and unified for all devices. If we have a working web application that’s sufficient for initial stage and can be considered as a product. Also, if we have a working web application, we can extract as APIs and can be used easily to integrate with our mobile application, therefore this can be considered to be a part of phase 1. |
| 4 | In-App messaging for Doctors and Patients | Communication between doctors and patients can help a lot and make good customized support for patients when they discover their infection and how they prevent the spread of it to the people around them. |

# MVP User Story Justification

*Instructions: Pick one user story in the MVP and justify why that user story belongs in the MVP. Then pick one user story that is* ***not*** *in the MVP and justify why that user story doesn’t belong in the MVP.*

|  |  |  |
| --- | --- | --- |
| **Included in MVP** | **User Story** | **Justification for Including or not including the user story in the MVP Plan.** |
| Yes | As a CDC Official I can view the number of patients that have reported testing positive so that so that I can analyze the outbreak. | It’s logical that the main reason of implementing our application to monitor the virus spread, and provide support with unfortunate infected cases. |
| No | As a Healthcare Company I can download sale/order report so that I can keep record of accounts. | This user story is only enabling healthcare to download the reports while if that is not fulfilled till MVP, He still can see the main functions without organizing them in report mode. |

# Sprint Scenario Responses

*Instructions: After completing the Release and MVP Plan, provide a simple explanation for adjustments that the team would likely take given the different unexpected sprint interruptions provided below.*

|  |  |  |
| --- | --- | --- |
| **Sprint** | **Scenario** | **Response** |
| 1 | One of the user stories has a dependency on another system that won't be available until the next sprint. What should the team do about this user story? | We will swap a user story from the next sprint that has the same number of points or nearby and will replace that with this user story. Flexibility and Improvisation is an important aspect of Agile and this can be implemented here clearly. |
| 3 | A software engineer is sick and will miss most of the sprint. What should you and the team do to account for their absence? | We will arrange a temporary resource as soon as possible and try to divide the workload between the team in such a way that when the software engineer is back from his sick leave he can easily cope up with the leftovers (if any). |