**Project Planning Phase**

**Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)**

|  |  |
| --- | --- |
| Date | 10 November 2022 |
| Team ID | PNT2022TMID54282 |
| Project Name | Classification Of Arrhythmia By Using  Deep Learning With 2-D ECG Spectral  Image Representation |
| Maximum Marks | 8 Marks |

**Product Backlog, Sprint Schedule, and Estimation (4 Marks)**

Use the below template to create product backlog and sprint schedule

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement (Epic)** | **User**  **Story**  **Number** | **User Story / Task** | **Story**  **Points** | **Priority** | **Team Members** |
| Sprint-1 | Download The  Dataset | USN-1 | We can download the Dataset contains Six classes | 4 | Low | Mathangi sanjana |
| Sprint-1 | Import The  ImageDataGenerator  Library | USN-2 | We can import ImageDataGenerator | 4 | Low | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-1 | Configure  ImageDataGenerator  class | USN-3 | We can configure the  ImageDataGenerator class | 6 | Medium | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-1 | Apply the  ImageDataGenerator | USN-4 | We can apply ImageDataGenerator to train dataset | 6 | Medium | Koneti lizitha  Kiruthiga.M  Madhumitha.M |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement (Epic)** | **User**  **Story**  **Number** | **User Story / Task** | **Story**  **Points** | **Priority** | **Team Members** |
|  | functionality to  Trainset and Dataset |  |  |  |  | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-2 | Import Libraries | USN-5 | We can import required Libraries | 1 | Low | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-2 | Initialize the Model | USN-6 | Initializing the Image recognition model | 1 | Low | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-2 | Adding CNN layer | USN-7 | We can add Convolutional Neural Network(CNN) used for image/object recognition and classification | 2 | High | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-2 | Adding Dense Layer | USN-8 | We can add Dense Layer in which each neuron receives input from all the neurons of previous layer | 4 | High | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-2 | Configure The  Learning Process | USN-9 | We can configure The Learning process which is a method, mathematical logic or algorithm that improves the network's performance and/or training time. | 4 | High | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-2 | Train the Model | USN-10 | We can train our model with our image dataset. fit\_generator functions used to train a deep learning neural network | 4 | High | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-2 | Save the Model | USN-11 | We can save The model with .h5 extension | 2 | Medium | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-2 | Test the model | USN-12 | We can Test the model through Loaded necessary libraries, the saved model | 2 | Medium | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| **Sprint** | **Functional**  **Requirement (Epic)** | **User**  **Story**  **Number** | **User Story / Task** | **Story**  **Points** | **Priority** | **Team Members** |
| Sprint-3 | Create Html files | USN-13 | We use HTML to create the front end part of the web page. | 8 | High | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-3 | Build Python code | USN-14 | We build the flask file ‘app.py’ which is a web framework written in python for server-side scripting. | 8 | High | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-3 | Run the App | USN-15 | We can run the App | 4 | Medium | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-4 | Register IBM Cloud | USN-16 | We can register IBM Cloud | 8 | Medium | Koneti lizitha  Kiruthiga.M  Madhumitha.M |
| Sprint-4 | Train the model on  IBM | USN-17 | We can Train Out model on IBM | 12 | High | Koneti lizitha  Kiruthiga.M  Madhumitha.M |

**Project Tracker, Velocity & Burndown Chart: (4 Marks)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story**  **Points** | **Duration** | **Sprint Start**  **Date** | **Sprint End Date**  **(Planned)** | **Story Points Completed (as on Planned End Date)** | **Sprint Release Date**  **(Actual)** |
| Sprint-1 | 20 | 5 Days | 24 Oct 2022 | 28 Oct 2022 | 20 | 28 Oct 2022 |
| Sprint-2 | 20 | 5 Days | 30 Oct 2022 | 04 Nov 2022 | 20 | 04 Nov 2022 |
| Sprint-3 | 20 | 5 Days | 06 Nov 2022 | 11 Nov 2022 | 20 | 11 Nov 2022 |
| Sprint-4 | 20 | 5 Days | 13 Nov 2022 | 18 Nov 2022 | 20 | 18 Nov 2022 |

**Velocity:**

To calculate the team’s **average velocity (AV)** per iteration unit

Where, 𝐴𝑣 = 𝑆𝑝𝑟𝑖𝑛𝑡𝑉𝑒𝑙𝑜𝑐𝑖𝑡𝑦 𝑑𝑢𝑟𝑎𝑡𝑖𝑜𝑛

**Average Velocity** - Story points per day

**Sprint duration** - Number of days (Duration) for Sprints

**Velocity** - Points per Sprint

Average Velocity is **4** points per Sprint 𝐴𝑣 = 205 = 5

**Burndown Chart:**

A burndown chart is a graphical representation of work left to do versus time. It is often used in agile [software development](https://www.visual-paradigm.com/scrum/what-is-agile-software-development/) methodologies such as [Scrum](https://www.visual-paradigm.com/scrum/scrum-in-3-minutes/). However, burn down charts can be applied to any project containing measurable progress over time.

